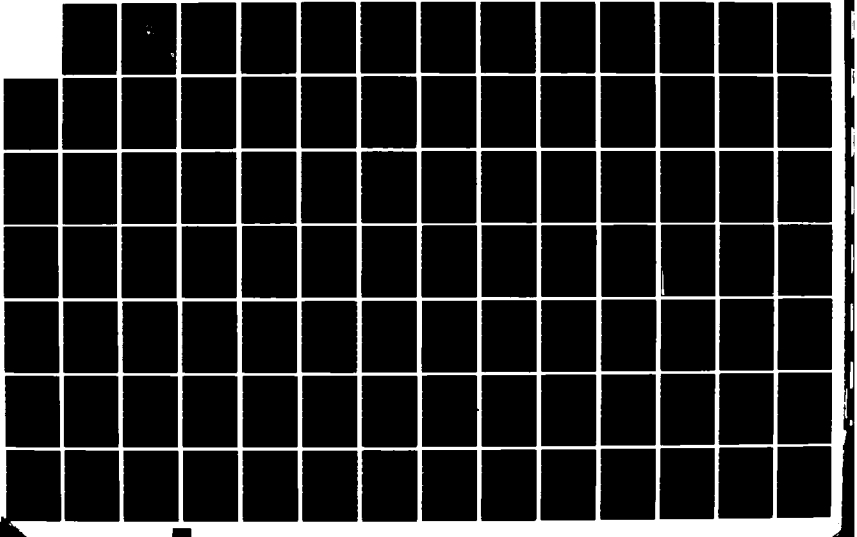
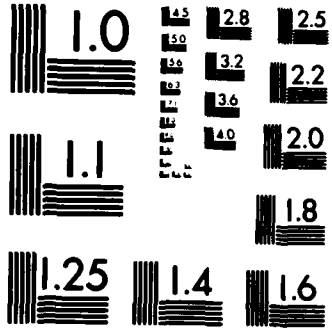
AD-A150 611 GENERAL DESIGN CONSIDERATIONS OF AN AIR FORCE INFORMATION SYSTEM(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA E AYTACER JUN 84 1/2

UNCLASSIFIED F/G 9/4 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

GENERAL DESIGN CONSIDERATIONS OF AN AIRFORCE INFORMATION SYSTEM

by

Engin Aytacer, Jr.

June 1984

Thesis Advisor: N. Lyons

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A150611	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) General Design Considerations of an AIRFORCE Information System		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis June 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Engin Aytacer, Jr.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE June 1984
		13. NUMBER OF PAGES 119
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Information Systems, Database, Computer Network, Tactical Air Force,high-end minicomputers, network topology, modularization requirements, system hierarchy, network architecture

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

General design issues of an Airforce information system are considered in this thesis. Current structure of the system is presented with its requirements. Information storing, retrieving and updating procedures are presented. And an example of a logical database is designed. Networking issues are expressed in an undetailed way. Finally, a set of high-end minicomputers are evaluated to present approximate cost of system. And a general methodology for minicomputer selection process is presented.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Approved for public release; distribution unlimited.

General Design Considerations of an AIRFORCE Information System

by

Engin Aytacer, Jr.
1St.It. ,Turkish Air Forces
B.S., Turkish Air Forces Academy, 1980

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1984

Author: F. Aytacer

Approved by: Norman R Lyons
Thesis Advisor

F. R. Johnson
Co-Advisor

David K. Hsiao
Chairman, Department of Computer Science

Kneale T. Marshall
Dean of Information and Policy Sciences

ABSTRACT

General design issues of an Airforce information system are considered in this thesis. Current structure of the system is presented with its requirements. Information storing, retrieving and updating procedures are presented. And an example of a logical database is designed. Networking issues are expressed in an undetailed way. Finally, a set of high-end minicomputers are evaluated to present approximate cost of system. And a general methodology for minicomputer selection process is presented.

Additional keywords: computer networks, topology, architecture, modules (electronics), interfaces, cryptography.

TABLE CF CONTENTS

I. INTRODUCTION 10

II. REQUIREMENTS ANALYSIS 12

A. DEFINITICN OF THE PROBLEM 12

B. SYSTEM ELEMENTS 12

C. HOW CURRENT SYSTEM WORKS? 14

1. Operation And Training 14

2. Personnel 15

3. Intelligence 15

4. Logistics 15

D. NEEDS 16

1. Generation Of Reports 17

2. General Bookkeeping 17

3. Messages And Orders 17

4. Schedules 17

5. Personnel Records 18

6. Intelligence Records 18

7. Logistic Records 18

8. Others 19

III. BREAK THE SYSTEM DOWN INTO MODULES 20

A. CENTRAL DATA,DISTRIBUTED DATA 20

1. Centralized Responsibilities 21

2. Decentralized Responsibilities 22

3. Types Cf Objectives For Distributed Data 23

B. SPLIT THE DATA 24

1. Geographical Division 24

2. Divisicn By Type Of Data 25

3. Division By Type Of Usage 25
C. SYSTEM CONFIGURATION 26
D. MULTIPLE COPIES OF DATA 28
1. Master Copy Updating 28

IV. DATABASE DESIGN 32
A. INTRODUCTION 32
B. SURVEY OF PROMINENT DATABASE MODELS 33
C. LOGICAL DATABASE DESIGN 34
1. Outputs Of Logical Database Design 35
2. Design Reviews 36
3. Logical Design Primitives 37
4. Primitives In The Real World 37
5. Primitives In The Conceptual World 38
D. THE SEMANTIC DATA MODEL 39
1. Defining Entity Classes 39
2. Nonbase Entity Classes 41
3. Defining Attributes 41
4. Attribute Value Classes 42
5. Additional Attribute Characteristics . . . 42
6. Defining Relationships 42
E. WEAPONS 45
F. AMMUNITION 46
G. PERSONNEL 48
H. INTELLIGENCE 51
I. STATUS 52
J. INSTALLATION 54
K. LOGISTICS 55
L. MAINTENANCE 57
M. COURSES 58
N. STRING TYPES 59

V. NETWORK CONSIDERATIONS 67
A. INTRODUCTION 67

B. NETWORK GOALS 67
C. NETWORK STRUCTURE 70
1. Point-To-Point channels 70
2. Broadcast Channels 72
D. NETWORK TOPOLOGY 72
E. DESIGN ISSUES 74
F. NETWORK ARCHITECTURES 75
1. The Physical Layer 77
2. The Data Link Layer 78
3. The Network Layer 78
4. The Transport Layer 79
5. The Session Layer 79
6. The Presentation Layer 80
7. The Application Layer 80
8. Line Control 82
9. Code Selection 82
10. Message Header 82
11. Control Messages 85
12. Compaction 87
G. ROUTING 87
1. Static Routing 88
H. TRANSMISSION ERRORS 89
1. Error Correcting Codes 89
2. Error-Detecting Codes 91
I. CRYPTOLOGY 92
1. Substitution Ciphers 93
2. Codes 96
3. Transposition Ciphers 97
4. The Data Encryption Standard 98

VI. COST CONSIDERATIONS 101
A. PRELIMINARY CONSIDERATIONS 101
1. Defining Needs 101
2. Throughput 102

3. User Requirements 103
B. THE HARDWARE CONFIGURATION 103
1. Communications 104
C. THE SOFTWARE QUESTION 105
D. THE SERVICE FACTOR 105
E. A MODEL FOR MINI COMPUTER SELECTION 106
F. A RIGOROUS EVALUATION MODEL 106
1. Data General Corporation, Eclipse MV/8000 II 107
2. Dec Vax-11/730 108
3. Ibm System/38 5381 Model 3 109
4. Prime Computer, Inc. Model 250-II 109
5. Wang Laboratories, Inc. VS 85 110

VII. CONCLUSIONS 116

LIST OF REFERENCES 118

INITIAL DISTRIBUTION LIST 119

LIST OF TABLES

1. Subjective Factor Weights 112
2. Computation Of Subjective Factor Measures . . . 113
3. Summary Of Objective and Subjective measures . . 114

LIST OF FIGURES

2.1	System Hierarchy Chart	13
3.1	Hierarchical Dependent Data Configuration	27
3.2	Modular Structure Of The System	31
4.1	Classification Of Database Models	33
4.2	Flow Of The Database Design	34
4.3	Conceptual Primitives	38
4.4	Entity Class Description	40
4.5	Attributes	41
5.1	Classification Of Interconnected Processors	69
5.2	Communication Subnet	71
5.3	Possible Topology Types	73
5.4	ISO Seven Layer Architecture	76
5.5	Cryptographic Method	93
5.6	Transposition Cipher	98
5.7	A: P-Box, B: S-Box, C: Cascade Ciphers	99
6.1	Subjectivity, Objectivity Relation	114

I. INTRODUCTION

An Air Force which has no automated processing capability can not accomplish its tasks, in a modern sense. The lack of an automated data processing capability will bring too many problems. The dynamic nature of the system can not be supported by an old fashion mechanical or semi-automated information gathering, storing, and retrieving methods.

New system will need a powerful processing capability and a communication media to exchange information between the geographically separated subparts. Establishment of a computer system will need a huge investment in terms of both hardware and software requirements. But the system can pay back itself by improving the speed, reliability, security and maintability of the processes.

First section addresses the current system structure and the requirements for the new system. Requirements are not discussed in detail. Instead they presents general system requirements.

Second section divides the system into modules. Since the modularization is an important design methodology, we also mentioned it in this part. Thus, now it is easier to implement the system module by module, in a structured way. We also set the interfaces among these modules.

In third section we introduced a database model. And an example of this model is presented.

Section four discusses the networking issues. Since our system needs a network to interconnect the independent hosts, we presented international standards organization's seven layer architecture model. Also, cryptographic methods are discussed in this section.

Section five addresses a minicomputer selection methodology rather than system's cost analysis. Since system cost can depend cn the market conditions, we preferred to present this methodology. Thus, it can be used with different computer products under different market conditions. And an example of the cost evaluation is presented.

II. REQUIREMENTS ANALYSIS

A. DEFINITION OF THE PROBLEM

An air force which is made up of two tactic air forces, an airlift command, one training command three supply, maintenance centers has no automated processing capabilities. The problem is to automate daily mechanical data retrieving, storing and updating the jobs. Thus efficiency, reliability and speed of the system will greatly be increased.The actual statement of the problem is design of an information system for an air force command.

B. SYSTEM ELEMENTS

As we mentioned earlier, this air force is an hypothetical one which has no strategic missions. Tactic forces are considered in the same structure. Each of them has four air bases and its own early warning report center. And again each base is considered to be in the same structure. Each base has two squadrons one material command and one general support command. At the same time each tactic force has its own supply,maintenance center. There is only one airlift command which is directly connected to the air forces command. Air training command is also directly connected to air forces command. This simplified general structure of the system is more convenient to be designed easily. Further extensions can be made easily after completing the design of this more general system. In fact, expandability is the one of the system design aspects. The detailed hierarchy chart of the system is shown in figure 2.1 .

Early warning centers have their own CRP's(control report post). They control the activities of the friendly

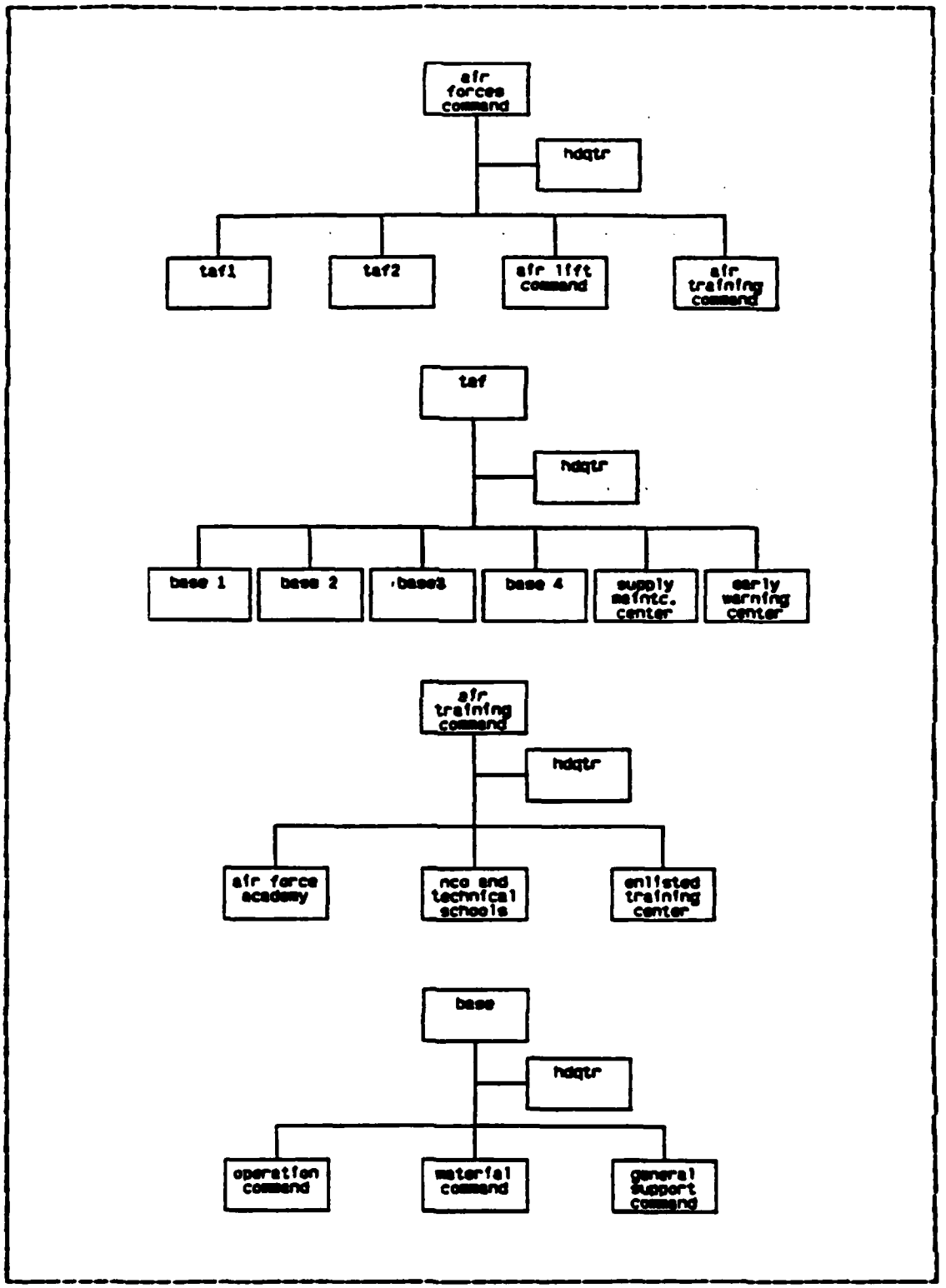


Figure 2.1 Systems Hierarchy Chart.

and enemy forces in their responsibility region. Especially they need accurate and fast information system, to inform the friendly forces at right time. Also supply and maintenance centers are the backbones of the system. The response for a particular demand sometime may take a long time. To decrease this time, local processing power is not sufficient. They also need a computer communication with the other centers. This system especially will payback itself in these two areas.

C. HOW CURRENT SYSTEM WORKS?

Before considering system requirements we should first check the current system environment. Air operation center claims daily status reports from the TAF and airlift command. These reports are related with the current operational readiness of troops. Early warning centers send these reports to air operation center of the command. These reports are different from the others in a sense that they have to be updated immediately. In EWC environment each report post has to send status information to its related EWC. All troops, have to report all emergency situations immediately to the upper level commands. The general structure of the headquarters are the same except the size of them. Each headquarters has four main parts. They are operation, training, intelligence, logistics and personnel. We will consider this structure in the same way when we implement them. Main responsibilities of these systems can be summarized in the following way.

1. Operation And Training

This part of the system is responsible for the operational readiness. Administration of operations and exercises are the main responsibility of them. Training the personnel for operational readiness is another important

assignment of this part. Planning the operations and exercises are also their responsibility.

2. Personnel

This part of the system is responsible to keep all personnel records: promotions, health status, training and criminal background, assignments, military home status. Also this part plans the future personnel policy. It determines the training needs for new type of jobs and offers new courses related with the new systems.

3. Intelligence

This part of the system related with the enemy status. They plan the intelligence needs and provide them. After providing the needs, this intelligence is evaluated and distributed to related troops. They also interrogate the pilots who are completed the mission. Preparing the necessary maps and target files is also their responsibility.

4. Logistics

This part of the system is responsible for providing the every kind of material and stocking, distributing them in an intelligent way. Maintenance activities are also their responsibility. To accomplish these activities it has three regional supply and maintenance center. Logistic planning is the most important responsibility of this part.

These four main part have their own subdivisions to accomplish their task effectively. These subparts are: flyer, navigator, maintenance, supply, civil engineering, electronic weapons, ammunition, communication, transportation, early warning, anti-aircraft and missile, and comptroller.

D. NEEDS

In this system environment, connections among troops and administration are so frustrating. We need to eliminate waste time which is spent on demand, supply, record keeping, processing, reports. First we have to speed up these activities and integrate them in a central authority which manages all these activities instantly. Also lower management needs its own processing power to keep his own activities, records. By providing automated processing system, lower level management can easily communicate with upper levels or vice versa. To provide something from upper management, low level managers need not to go laterally, in system hierarchy. They should have as little responsibilities as possible. The only level that they have to communicate with it is one higher level management. By the way higher level managers can easily control what is going on in lower levels. Daily reports have to be generated in a way that is accurate and fast. Message traffic in both direction have to be speeded up. Personnel do not have to be too much busy because of the daily routine jobs. The number of personnel who is in charge of doing these jobs have to be decreased. At the same time we should forget the papers which are in useless form in terms of processing. Office automation is the one of the important issues.

By providing processing power to lower level management, they can manage the jobs independently. Thus they can utilize all the sources efficiently, no overhead for higher level management. Quick response for a demand is the essential point for efficiency of operation. Personnel need more time for doing real jobs rather than doing daily routine bookkeeping things. If we would like to define requirements more precisely, we can summarize them in the following way:

1. Generation Of Reports

These reports are divided into two groups. One of them is periodicals. These reports are sent daily, monthly or semi annually. They contain personnel and material status informations. Other group is the nonperiodicals which are almost related with the emergency situations and special cases. They are usually urgent and have priority.

2. General Bookkeeping

At different levels there is a big necessity for bookkeeping activities. These are very large jobs and need frequent processing. In mechanical processing most of the time is spent to store and retrieve these information, rather then processing them. Storing them in mechanical way needs too much space and personnel time.

3. Messages And Orders

Sending and receiving messages are done by the means of teletypewriters. even in this environment this automation does not satisfy the system constraints.They are time consuming too. Some messages or orders have priority and secrecy. Storing and retrieving these messages are also time consuming. Nowadays message traffic is over the standard level. Electronic mailing is the one of the aspects of the new system.

4. Schedules

There are different kind of scheduling requirements in different units. All these scheduling activities are done by manpower. As a result of these schedules are not productive, economical, even sometimes they are big clashes. Lack of the local computer processing power, results in a time consuming, unproductive scheduling.

Especially airlift command needs different scheduling algorithms and its own processing power.

5. Personnel Records

Personnel records consist so much information about personnel. Decisions on personnel are made according to these records. Storing, retrieving, searching of these records are very hard and time consuming. Promotions, assignments are also made according to these records. Each year all records are searched all together. By mechanical way all these activities take a year long. Payroll system has been already set up. For other tasks there is a need for database system to process all these activities.

6. Intelligence Records

These records like personnel records are continuously changed and kept in files. We would like to access these records frequently. It will be very useful to keep them in a database in terms of storing, retrieving, updating, distributing them easily.

7. Logistic Records

Probably the largest manpower is consumed by keeping the logistic records there are millions of items in inventory of the air forces. Stock control is an important issue. Keeping the names and part no's of these items and location of them is the most time consuming problem. We need a very big processing capability and a database. Also the nature of the problem requires a computer network which is established among the supply centers and the related commands. The slowest part of the entire system is the logistic system. Also supply centers need a computer processing power and a database to keep their records.

8. Cthers

Air forces need a computer network to accomplish its task in a modern sense. Since the subparts of the systems are related with each other, computer network is the one of its requirements. All the jobs are in the form of stcring retrieving, updating or processiny of information. Thus we need a well designed database system to handle these jobs.

III. BREAK THE SYSTEM DOWN INTO MODULES

A. CENTRAL DATA,DISTRIBUTED DATA

Before considering the issue of how we can break the system down into modules, we should first look at the issue of what should be centralized and what should be decentralized. The answer to this question will determine the modular structure of the system. In the first chapter we have given the hierarchical structure of the system. System components are geographically distributed. According to system hierarchy and the system requirements we should answer the question above. The DP or DDP strategy should start with a top management view of where the corporation is going and how its direction might be changed by networks, database usage, microcomputers and DDP. The implemention of data processing takes one of two forms: designed or ad hoc. In ad hoc systems user groups do their own things hoping for no external interference. Designed systems need to begin with a corporate strategy. This strategy will keep us away from a dozen side effects of ad hoc implementation. When we were talking about the needs of the system; we have said that the local processing power is the one of the important issues.In distributed environment,the users can develop their own programs and now may have powerful compilers, report generators and other software on a central system to help them. While most of their programs use the local machine, they are not restricted to using only that. They can develop applications which sometimes need the power of large remote machine or data which are maintained centrally. Centralized standardization can be welcomed by decentralized groups if it has the appearance of being a service to them, as in the case of a corporate data dictionary.

Distributing processing can thus facilitate local autonomy, initiative and responsibility without obliterating the advantages of centralized data planning and data-base software. The user departments are freed from most of the frustrations of being tied to an overburdened DP department. They can achieve much quicker implementation of their requirements and tailor them exactly to their needs . They can quickly modify their programs to adjust the local situations. The rigidities of the centralized scheduling are avoided. The centralized authorities on the other hand have not lost the ability to manage the overall system's growth in the corporation. There is a delicate balance between the degree of local autonomy and the degree of centralized control. We can separate the local and the central responsibilities in the following way:

1. <u>Centralized Responsibilities</u>

    1. definition of local and central responsibilities (design considerations)
    2. Choice of network standards
    3. Choice of data descriptions language
    4. Choice of data-base software
    5. Database administration services including the coordinated design of the data
    6. Maintenance of a system data dictionary
    7. Selection and design of applications to serve multiple locations
    8. Selection of applications to be transferred between locations

9. Review of documentation of applications transferred between locations

10. Guidance cn modularization needed to facilitate applicatien transfer

11. Technical consulting services

12. System security design and administration

13. Design of system auditing controls and facilities

2. Decentralized Responsibilities

1. Local application development
2. Design of locally used files
3. Design of subschemas which relate to a remote database in conjunction with a central database administrator
4. Selection cf equipment within the constraints of corporate recommendations
5. Development and documentation of applications intended for transfer to other locations
6. Modifications of applications received from other locations
7. Liaison with centralized authorities on schema development and standardization of data items
8. Liaison with other locations on applications developed for common use

In our system design,distributed is the one of the design issues. So we should also give some objectives which justify our design ccnsiderations.

3. Types Of Objectives For Distributed Data

1. A highly distributed user community
2. Low cost. Data replication is cheaper than long-distance transmission for the traffic volumes. This argument is stronger for countries which do not have low-cost data networks.
3. A need for high availability the data remain available when one or more copies of it are inaccessible or long distance transmission links are down.
4. The military need for surviveability. Data remain available after destruction of multiple nodes.
5. Fast response time. Access to a local data is faster than access to a distant highly shared data.
6. Tunability. Data can be moved to different nodes as usage patterns change. Data heavily used in one geographic region can be stored near that region.
7. Traffic volumes are too high for a single storage system.

A major task for more corporations over the ten years ahead is to decide what databases they need, where they are best located, what data should be stored in them and how they should be organized. The amount of data stored will increase drastically, and the ways the data are organized will be fundamentally changed to increase their usefulness. Data bases will become the foundation stone of much corporate data processing. One of the most difficult

tricks that we have tc learn is how to introduce automation without introducing rigidity. Database techniques are an important part of the answer. A database is intended tc make data independent of the programs that use them. Old application programs do not have to be rewritten when changes are made to data structures, data layout or the physical devices on which data are stored. The data can be easily reorganized or their structure added. Now with the distributed systems we have a new requirements. If the data are distributed existing programs should not have to be rewritten. In other words we want the data-base system itself to be distributed.

B. SPLIT THE DATA

Data can be divided within a distributed system as fcllcws:

1. Geographical
2. Type of data
3. Type of usage

1. Geographical Division

If data originate and are used in given geographical areas then geographical division may make sense. This type of divisicn can;

1. reduce tle cost
2. Increase the system availability
3. Increase the accessability of the data
4. Give faster response time
5. Permit lccal users to maintain control over their own data

On the other hand it may;

1. Increase the total system cost
2. Become ccmplex

3. Be unsecure
4. Be very difficult to integrate later

2. Division By Type Of Data

In large organization computer systems perform different sets of functions. They may be entirely different independent systems. However, telecommunications links are found between them, or one terminal may have access to multiple systems. The data are divided by data type. The advantages of this type division can be;

1. Local implementation and control are desirable
2. Simplicity. To put all applications on one machine can be highly complex. It is often easier to implement smaller systems.
3. Politics. Local management wants control of its own data processing.
4. Security. The data could be reconstructed at the other center from log tapes and file dumps.

The major danger of of separate data systems arises when data are designed in incompatible ways. Many corporations, government departments and military organizations have staggering proliferation of incompatible data and a growing need to merge data or develop applications which use data from separate systems. In most cases the desired conversion or migration is never performed.

3. Division By Type Of Usage

Sometimes the distinction is made according to the usage of the system. If it is a production or an operation system, it should be designed for a precisely defined set of operations. The exact nature of the system is known in

advance. In an information system the nature of the queries is not known in detail. The data structures are designed to handle spontaneous queries which may differ widely in their nature and may require the data to be searched in various ways. The value of a good information system is sometimes higher than that of an operations system using the same data, because it enables management to make better decisions.

Our system will be geographically divided,because of the nature of the system needs this type of division. After these considerations, now we can apply all of them in our system. First of all we should determine the system configuration.

C. SYSTEM CONFIGURATION

Since our system has a hierarchical structure and the nature of the communication is vertical we should choose a configuration which satisfies our goals which are discussed in chapter 2. Hierarchical configuration is most appropriate one to satisfy our goals. The data in lower-level machines are closely related to those in the higher-level machines. They are often a subset of a higher-level data used for applications. The master copy of the data may be kept by the higher-level machine. When a change is made to the data in the lower level machine, this change must be passed up to the higher level machine sometimes immediately, sometimes later in updating cycle. This is the dependent hierarchical data configuration which has been shown in Figure 3.1 . In our implementation sometimes higher-level management need not to have all the data reside in lower-level machine, or lower level machines may store the some of the data which are in the higher machine and also have some which are its own they are never passed upwards. The bulky data are never

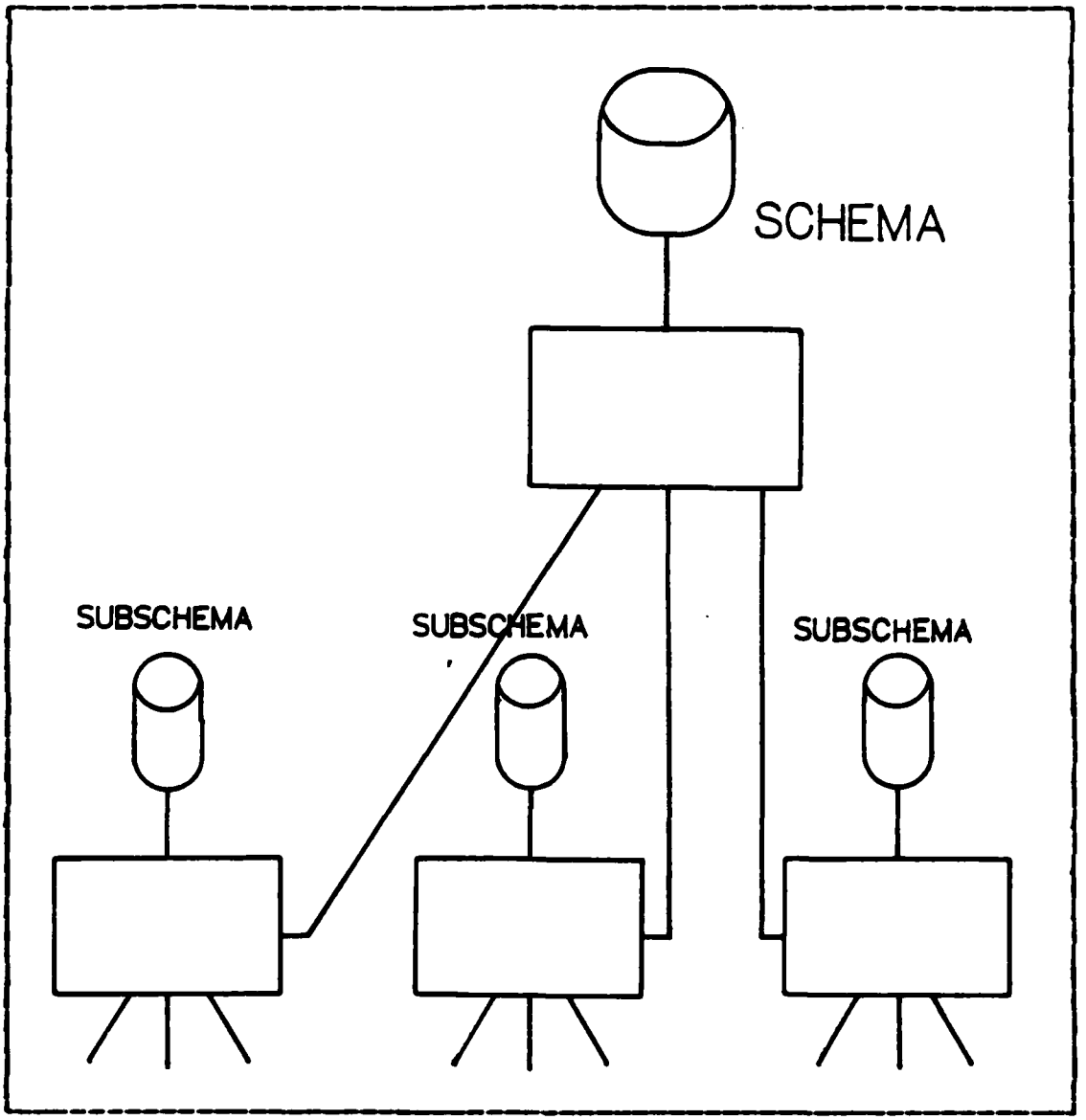


Figure 3.1 Hierarchical Dependent Data Configuration.

needed by the higher-level parts. For instance Air Force Command only need to have the number of operational or unoperational aircrafts. It may not need to know what kind of problems that these aircrafts have. The detailed data related to these aircrafts might be kept in base's database. On the other hand, personnel files are kept by both of the Command and base itself.

D. MULTIPLE COPIES OF DATA

In our implementation some components will have the same data. In this case care is needed in planning the updates and the recovery from failures. Restart and recovery can present problems if the database are being continuously updated. The most easily controllable approach is to have single copy of the data. The other replicated are regarded as secondary to the master copy. The system is designed so that if the master copy is destroyed it can be reconstructed.

Different data each has a single master copy but these could be stored in different locations. In our system implementation Air Force Command will have the some parts of the database of TAFs, Airlift Command, and Training Command. And TAFs also will have the some parts of the database of the bases. If anyone of the AFCOM's or TAF's database is destroyed, integrity can be provided from the copies of data which are at lower-level command's databases. This application is neither exactly master copy approach nor freely distributed approach but the combination of them. This application will bring us the benefits of the both approaches. The most important one is the storage saving. By storing the only some part of the data of lower level command's databases will save us a lot of memory space. And by providing the recovery procedures will provide the database integrity.

1. Master Copy Updating

When we use the master copy approach database may or may not be updated in real time. There are two approaches.

1. All transactions immediately update the master. The master issues new copies of changed records to other processes periodically.

2. Transactions update a nonmaster file. All transactions are saved for periodic updating of the master. When the master is updated new copies of changed records are sent to other processors which use them.

After a failure of part of the system resynchronization is achieved by issuing new copies of any changed records in the master to the processors which keep them. If the master itself fails then copies of the transactions must be kept until it recovers. So that it can be updated and then in turn issue copies of the changed records to other processors which keep them.

In our implementation the only real time updates are the Status Reports updates and EW updates. All the other updates are saved and then they periodically update the master.

Now we can break the system down into modules according to considerations presented so far. AFCOM has its own mainframe computer. It also includes the data which are in TAF's, Supply and maintenance center's, Airlift Command's and Air Training Command's database. TAFs hold some data which come from the bases. Base is the smallest module which gathers information from its subparts. TAFs also hold the same data which come from the its own supply and maintenance center. The most important data TAFs have, is the early warning reports. Those data need to be updated in real time, and have to be sent to command's database. Daily routine status reports are sent to TAFs database. Those reports are also updated in real time, since they are very small and important in terms of operational decision making. All other kind of data can be updated periodically.

In system hierarchy lower-level modules can reach to top level via the intermediate level modules. They can't

access to top level independently. TAF 1 and Base 1 are located at the same place. TAF 2 and Base 5 are located at the same place. One of the supply and maintenance center is located at the AFCOM's location. These are important only in terms of hardware requirements. In base environment subparts have terminals which provide communication with base's mini computer. TAFs, ATCOM, ALCOM and bases have their own minis. In general each level communicate with one level higher. For instance bases first have to update the TAF's database.Then gathered updates can be transmitted to Command's database by TAFs. This will reduce the communication costs. In urgent situations a module can immediately update the both Command's and TAF's database. For instance early warning reports and status reports can be updated parallelly. Our system has the same structure which has been shown in Figure 3.1 . This kind of configuration will eliminate the complexity. If every module of the system interacts with every other the number of interactions grows at approximately the square of the number of modules N(N-1)/2 interaction for n modules. It has been observed that as system grows they become more complex and harder to manage. The complexity and cost following roughly a square law. The optimum solution can be found at a point in which economies of scale and complexity are balanced. [Ref. 1] In this environment the system modularization can be demonstrated in Figure 3.2 .

In this implementation we considered two things. First of all geographically close nodes connected to centers. And those connected nodes are also hierarchically related with each other. Thus this implementation is more appropriate than the others, like pure logical connection or pure geographical connection. Also it is convenient to be modified easily whenever we need to change the system configuration.

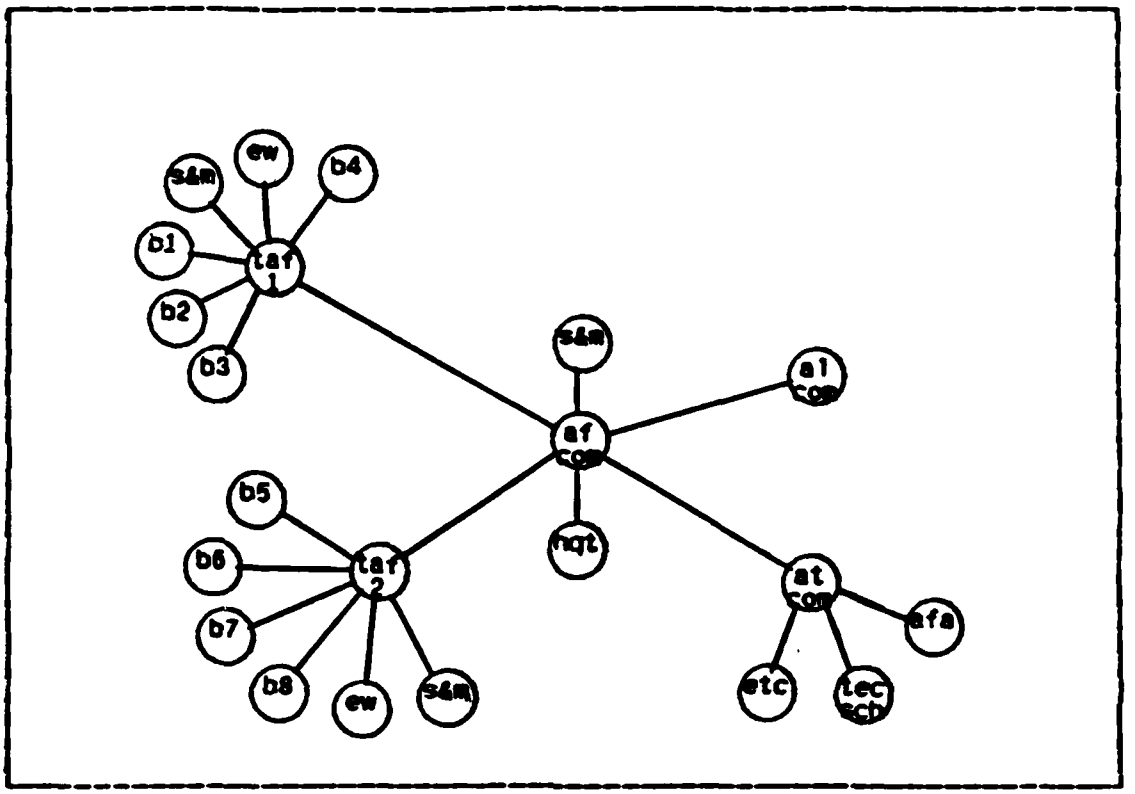


Figure 3.2 Modular Structure Of The System.

IV. DATABASE DESIGN

A. INTRODUCTION

A database is the interface between the people and machines. The nature of these components is utterly different. The difficulty is to develop a database which meets the needs of the people who will use it, and which is practical in terms of technology and hardware. Since the database is the bridge between humans on one side and hardware on the other, it must match the characteristics of each.

A database supports a community of users whose needs partly overlap, partly diverge, and partly conflict. Unfortunately there is no algorithm for database design. Database design is both art and science. Dealing with people understanding what they want today, predicting what they will want tomorrow, differentiating between individual needs and community needs and making appropriate design tradeoffs are artistic tasks.

Database design is a two-phased process. First we examine the users's requirements and build a conceptual database structure that is a model of an organization. This phase of database design is often called logical database design. Once the logical design of the database is completed, this design is formulated in terms of a particular DBMS . Usually compromises must be made . For example the DBMS may not be able to express relationships precisely as the users see them. The process of formulating the logical design in terms of DBMS facilities is called physical database design.

We have examined the user requirements in Chapter 1. In this Chapter we will build a conceptual database structure, namely the logical database design.

B. SURVEY OF PROMINENT DATABASE MODELS

Figure 4.1, portrays six common and useful database models. Models on the left-hand side of this figure tend to be oriented toward machines and machine specifications. Two categories of database models have been omitted. These categories are the hierarchical and network data models.A model is hierarchical if its only data structure a hierarchy (tree). A model is network if its data structures are both trees and simple networks. Only complex networks need to be decomposed before they are represented. The hierarchical model has become too narrow and the network data model too broad.

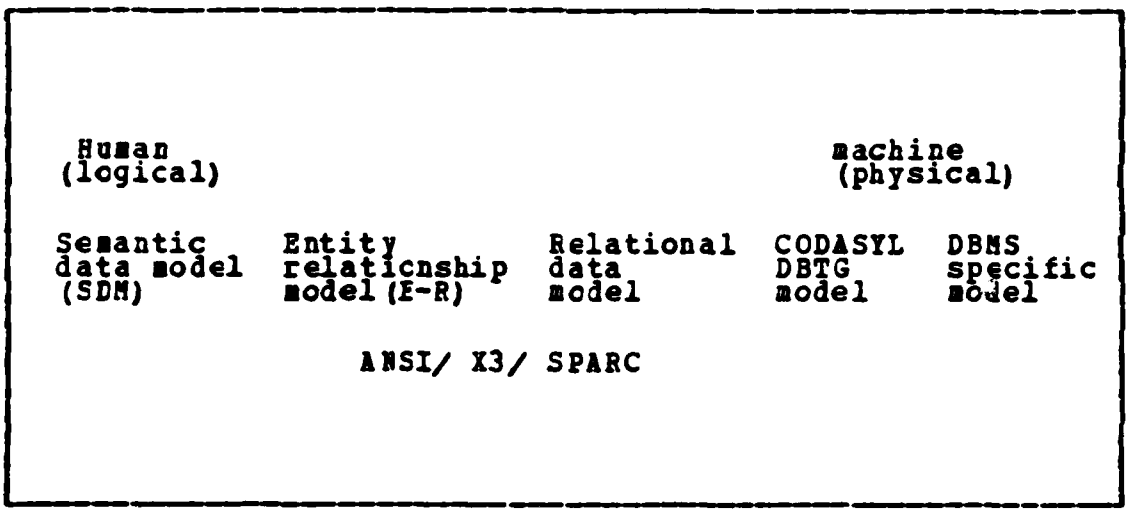


Figure 4.1 Classification Of Database Models.

C. LCGICAL DATABASE DESIGN

As stated database design is an intuitive and artistic process. There is nc algorithm for it. Typically database design is ar iterative process during each iteration, the goal is to get closer to an acceptable design.Thus a design

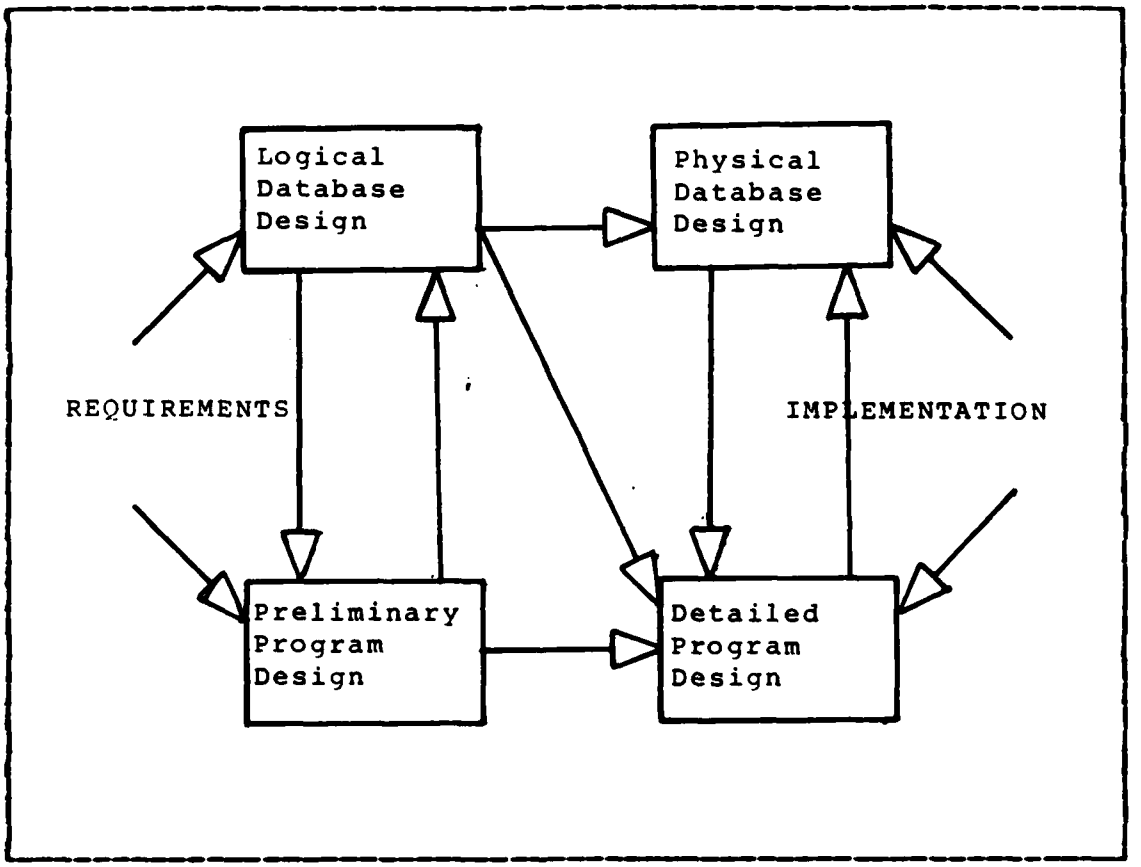


Figure 4.2 Flow Of The Database Design.

will be developed and then reviewed. Defects in the design will be identified and the design will be redone. This process is reviewed until the development team and users can find no major defects. Figure 4.2, illustrates the flow of work in a typical database design project. User requirements are studied and a logical database design is developed.

Concurrently the preliminary design of database processing, programs is produced. Next the logical database and the preliminary designs are used to develop the physical database design and the detailed program specifications. Finally both of these are input to the implementation phase of the project.

1. Outputs Of Logical Database Design

A logical database design specifies the logical format of the database. The records to be maintained, their contents and relationships among those records are specified. It is sometimes called the schema, the conceptual schema, or the logical schema.

a. LOGICAL DATABASE RECORDS

To specify logical records the designer must determine the level of the detail of the database model. If the model is highly aggregated and generalized, there will be few records. If the model is detailed there will be many records. The database designer must examine the requirements to determine how coarse or how fine the database model should be. The contents of these records are specified during logical design. Names of the fields and their format must be determined.

As the requirements are evaluated and the design progresses constraints on data items will be identified. These constraints are limitations on the values that database can have. Three types of constructs are common. Field constraints limit the values that a given data item have. Intrarecord constraints limit values between fields within a given record. Interrecord constraints limit the values between fields in different records.

b. LOGICAL DATABASE RECORD RELATIONSHIPS

The essence of database is the representation of the record relationships. These relationships are specified during the logical design. The designer studies the application environment examines the requirements and identifies the necessary relationships. In general the relationship can always be omitted later in physical design, where as if the relationship were omitted during logical design, it would be difficult to add it later. The determination of record types and relationships is an iterative process. While identifying relationships the team may discover a reason for a new record type; and while discussing record types the team may identify a relationship. These two constructs are designed simultaneously.

2. Design Reviews

The final stage of logical database design is a review. The logical schema and the user views are examined in light of the requirements and program descriptions. Every attempt is made to identify omissions, unworkable aspects, or other flows in the design. Typically a panel of independent data processing people is convened for this review. Documentation of the logical schema, user views, and program descriptions is examined by the panel and the oral presentations are evaluated. At the conclusion of the design review the panel produces a list of problems discovered and a recommendation regarding the next step to be taken. The panel usually recommends that the project be continued while identified problems are fixed. Occasionally however the design panel may recommend that the database design be repeated. Very rarely the panel may recommend that the project be discontinued.

3. Logical Design Primitives

A logical design is a representation of reality. It is a model of aspects of some activity that are interest to sponsors of the database. Such models represent selected portions of reality; the models are developed by aggregation and generalization. Since the database represents a subset of reality, some questions about reality become unanswerable. The goal when developing a database design is to make only uninteresting questions unanswerable.

4. Primitives In The Real World

Before we begin to design files and relationships we need to describe the foundations of the data modeling. We need to answer questions like: Where do we begin? What is it that we are trying to represent? With what aspects of reality are we to be concerned? We have to be careful answering these questions.

To make progress, therefore we need to establish a starting point. We will begin with a set of fundamental structures or primitives.

1. The first primitive is object. The real world has objects they are phenomena that can be represented by nouns.

2. Objects are grouped into object classes, another important primitive. This is done by generalization.

3. Objects have properties. A property is a characteristic of an object.

4. The collection of all possible instances of a property is called a property value set.

5. A fact is an assertion that for a given object, a particular property has a particular element from the property value set.

6. Objects can be related to one another. These relations are called associations. Associations can exist between objects of the same class or of different classes.

5. Primitives In The Conceptual World

When we design or process a database we are not working with real world primitives like objects and properties. We are working with the representations of these

Real world primitives	Conceptual primitives
Object	Entity
Object Class	Entity Class
Property	Attribute
Property value set	Domain
Fact	Value
Association	Relationship

Figure 4.3 Conceptual Primitives.

primitives. As we defined in previous subsection there are correspondences of the real world primitives which are called conceptual primitives which are illustrated in Figure 4.3 .

D. THE SEMANTIC DATA MODEL

The semantic data model (SDM) was developed by Hammer and Mcleod [Ref. 2] and first published in 1981. We will use this model to design our database.

The most important advantage of SDM is that it provides a facility for expressing meaning about the data in the database. During logical database design we need such a facility to avoid confusion and to document learnings, design decisions, and constraints. SDM provides better facilities for such documentation than other data models.

Another advantage of the SDM is that it allows data to be described in context. Users see data from different perspectives. They see it relative to their field of operation. SDM allows relative data definition.

A third advantage of SDM is that constraints on database data can be defined for example if a given item is not changeable SDM allows this fact to be stated. Also, if an attribute must conform to a particular format these considerations can readily be defined. With other data models such constraints are not part of the schema description and are documented separately.

SDM describes the structure of data instead of structure of programs. SDM has certain structures and rules and with those structures and rules the designer has a good deal of latitude and flexibility.

1. Defining Entity Classes

Entities are organized into classes. Each SDM entity class may have a name, a description, members (the entities) and two kind of attributes. Figure 4.4, shows the basic format of an SDM entity class description. In this figure entity class names are printed in capital letters. SDM terms are shown in small letters followed by a colon.

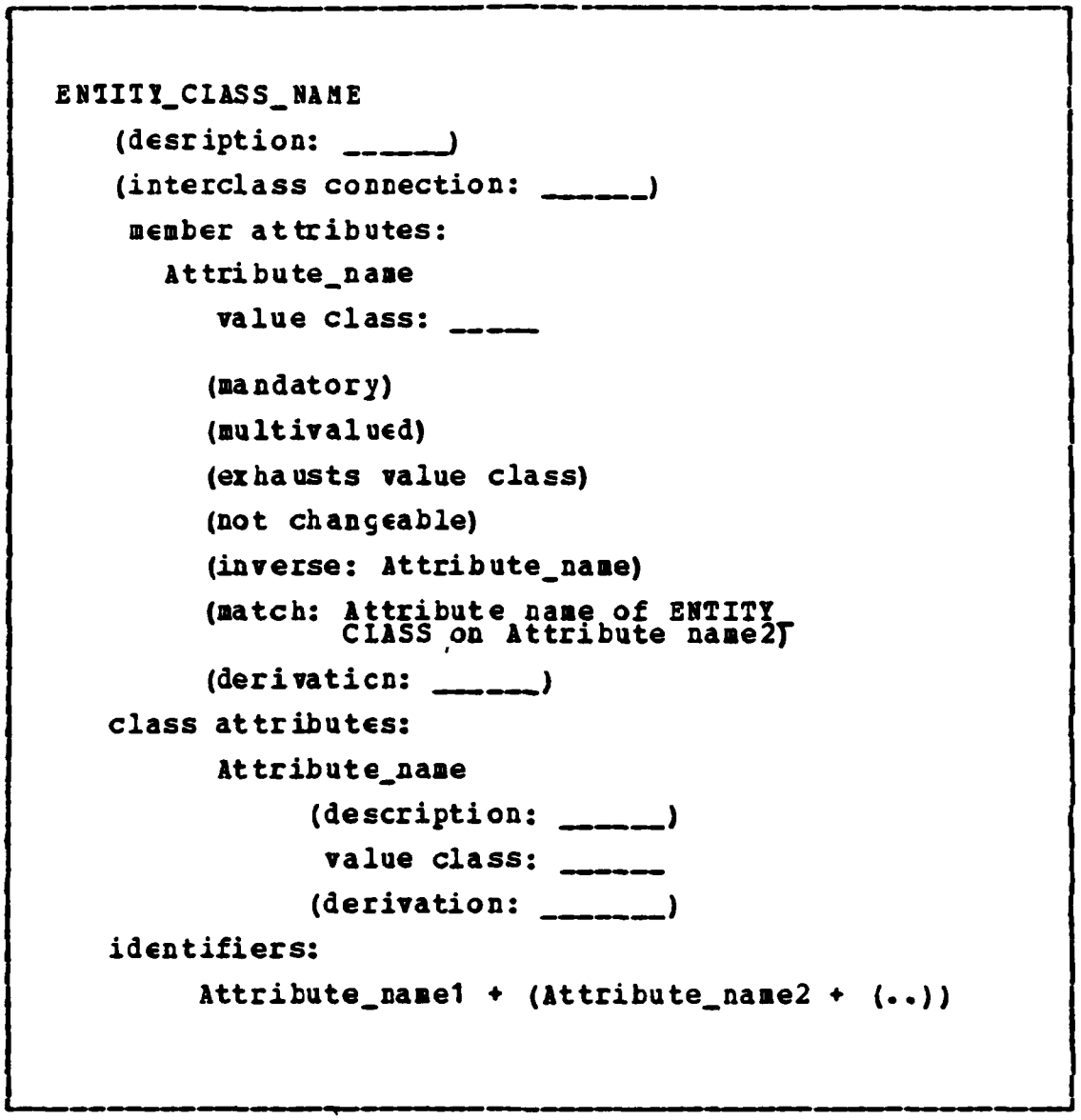


Figure 4.4 Entity Class Description.

Entity class is named and then an informal description of the class is provided. The description which is opticnal defines the purpose of and content of the class. Special remarks are also written here. Next the member attributes are defined. These are attributes of the entities in this class. Class attributes, which are optional belcng to the class as a whole and not to any particular member.

2. Nonbase Entity Classes

Nonbase classes are constructed from subsets of of other classes. Every nonbase class has an entry Interclass connection that describes how the class is to be constructed. For subsets the interclass connection names another class and specifies which members of that class are to be included in the new class.

3. Defining Attributes

Each entity or member of a class has a set of attributes. These attributes represent the properties of objects. In SDM each attribute has a name, an optional description a value class and a set of optional characteristics. These are

Name	Mandatory	Initial capital letter
Description	Opticnal	Remarks about attribute
Value class	Mandatory	The domain of the attribute

Descriptor characteristics	Default Value
Single or multivalued	Single
Value cptional or mandatory	Optional
Changeable or not changeable	Changeable
Exhaustive or nonexhaustive	Nonexhaustive
Overlapping or noncverlapping	Overlapping

Figure 4.5 Attributes.

summarized in figure 4.5 Attribute names are printed with initial capitals. They must be unique within the class where they are defined. Also, they must be unique within all classes that derived from their class of definition.

4. Attribute Value Classes

Each attribute must have a value class. In SDM value classes are defined as entity classes. Thus the definition of every attribute references another value class which references another and on and on. However regression is stopped in SDM by the existence of special class called Strings. String is a default entity class; it needs no definition. Strings contains any character string that the designer wants. Any nonambiguous definition of a string is allowed.

There are two advantages of defining attribute domains as entity classes. First the formats and the constraints on data items are clearly specified. Second allowing attributes to be members of entity classes permits a very natural, human oriented way of expressing the relationships.

5. Additional Attribute Characteristics

As we have shown in Figure 4.5, there are some additional characteristics of attributes.

Attributes can be single or multivalued (like repeating fields); they can be mandatory, meaning that a null value is never to be accepted; attributes can be not changeable, meaning that except to correct errors, the value of the attribute must remain same.

An attribute can be exhaustive, meaning that every member of the value class must be used. A multivalued attributes can be specified as nonoverlaping this means that a member of the value class can be used at once.

6. Defining Relationships

SDM provides three facilities for defining relationships. All three facilities use the SDM characteristic that entities can be contained within entities.

a. Defining Relationships With Inverse

The inverse facility causes two entities to be contained within each other. Physically this is impossible. Sequential including of each attribute to each other will result in infinite regression. But for our purposes we will ignore the infinite regression. After the first level all regressions are duplications of the data and hence unimportant. The beauty of SDM is that users can have it both ways. They can see data as they want to see data, even if these ways are seemingly contradictory. In actuality these views do not conflict at all. They are simply different views of the same thing.

b. Defining Relationships With Matching

The second SDM facility for representing relationships is matching. With matching a member of one entity class is matched with a member of another entity class. Then the value of an attribute in one of the members is moved to the other. The word ON defines how the members in the two classes must match.

c. Defining Relationships With Derivations

SDM provides a derivation capability for attributes. Derivation can be used to specify relationships among members in the same entity class. Thus we can derive new attributes from the existed ones. [Ref. 3]

So far we have talked about logical database design and the semantic data base model. Now,by using these information,we will design our database for Air Forces Command. First of all we should reconsider these requirements which have been discussed in Chapter 1. We should define the records, according to these requirements.

At the first step we built nine separate records for Air Forces Command. These records are in following:

1. Weapons
2. Ammunition
3. Personnel
4. Intelligence
5. Status
6. Installation
7. Logistics
8. Maintenance
9. Courses

These records make up the schema. Other levels in air force system have a subset of this schema. They have almost same records but the information in these records are limited with their needs.

For instance Tactical Air Forces have the information just for the bases which are dependent to TAF, Supply and maintenance centers, EW center. Bases can not have the information related with an another base. They have the information related only with its own subparts.

TAFs have exactly the same records to which AFCOM has. But the only difference, is AFCOM has much more data than bases have. In base's database courses record has not been included, because it is needless. Rest of the records are also included in base database. We designed only the minimal number of records here. Since, different applications will need different records. What we did here is an example of the logical database design. Thus, we will not consider the database needs of EWCs and supply, maintenance centers. These can be implemented in the same way which will be shown later in this chapter. Thus only one database design can be used by all the system components. This will be very effective in terms of simplicity, back-up, recovery, security and integrity. Especially this kind of design

consideration will be very useful, when we consider the network facilities for this system. Networking can be achieved easily because of the simplicity and regularity of databases which reside in different places and different machines.

After these considerations now, we can lock at the database itself. Again the formal structure of the database is taken from SDM, and it has the same structure with the one which has been illustrated in Figure 8.

E. WEAPONS

1. description: Includes all the information,related with every type of weapon systems which are currently in Air Force inventory.

2. member attributes:

   a) Wtype
      i) description:Type of the weapon system of the friendly forces and and enemy forces.
      ii) value class:WEAPON_TYPE
      iii) mandatory

   b) Wid
      i) description: Whether the weapon is enemy's or ours or both's.
      ii) value class: FRIEND_FOE
      iii) mandatory

   c) Wrange
      i) description: Effective range of the weapon system.
      ii) value class: RANGE

   d) Wfuel

i) description: Fuel capacity of the system if it uses fuel.

ii) value class: FUEL

e) Wlbs

i) description: Maximum ammunition load of the weapon system.

ii) value class: MAX_LOAD

iii) not changeable

f) Wammo

i) description: Types of ammunition which can be used by this type of weapon system .

ii) value class: AMMUNITION

iii) inverse: WHERE_USED

iv) multivalued

v) mandatory

g) Wnum

i) description: The total number of this type of weapon system.

ii) value class: NUMBER_OF_WEAPONS

iii) mandatory

iv) class attributes:

h) Wdate

i) description: Day of the year on which the information is valid(last changing date).

ii) value class:DATE

3. identifiers:WTYPE

F. AMMUNITION

1. descriptions: Includes all the information related with every type of ammunition which are currently in air forces inventory.

2. member attributes:

a) Acat
   i) description: Category of the ammunition which are used in weapon systems.
   ii) value class: AMMO_CATEGORY
   iii) mandatory
   iv) not changeable

b) Where_used
   i) description: Types of weapons which use this categcry of ammunition.
   ii) value class: WEAPONS
   iii) inverse: WAMMO
   iv) multivalued

c) Anum
   i) description: Total number of this category cf ammunition currently in inventory .
   ii) value class:NUMBER_OF_WEAPONS
   iii) mandatory

d) Albs
   i) description: Weight of one round of this categcry of ammunition.
   ii) value class: MAX_LOAD
   iii) mandatory
   iv) not changeable

e) Akill
   i) description: Killing radius of this category ammunition.
   ii) value class: RANGE

f) Awar
   i) description: Type of warhead of ammunition.
   ii) value class: WARHEAD_CAT

iii) multivalued

iv) class attributes:

g) Adate

i) description: Day of the year on which the information is valid (last changing date).

ii) value class: DATE

3. Identifiers: ACAT

G. PERSONNEL

1. description: Includes all the information related with the personnel who are in charge of Air Forces.

2. member attributes:

a) Pname

i) description:First name of personnel.

ii) value class:NAME

iii) mandatory

iv) not changeable

b) Plastname

i) value class: NAME

ii) mandatory

iii) not changeable

c) Paddress

i) value class: ADDRESS

ii) mandatory

d) Gradyear_degree

i) description: Graduation year from Air Force academy or technical schools and graduation degree are concatenated to make a unique key.

ii) value class: GRADYEAR_DEG

iii) mandatory

iv) not changeable

e) Assgnmntloc_date

i) description: Name of the locations at which the personnel have been there at least once, after graduation. Assignment date to those locations is concatenated.

ii) value class: LOCATIONS

iii) mandatory

iv) multivalued

f) Rank

i) value class: RANK

ii) mandatory

g) Prodate

i) Description: Date of the last promotion.

ii) value class: DATE

iii) mandatory

h) Specode

i) description: Indicates the personnel's special training or skill area.

ii) value class:SPECIALTY

iii) mandatory

iv) multivalued

i) Marsta

i) description: Marital status of personnel.

ii) value class: MAR_STA

j) Wife_name-occupation

i) desccription:Wife's name and occupation are concatenated .

ii) value class:NAME_OCCUPATION

k) Child_name_school

i) description:Child's name and school grade are concatenated . School grade is represented only with its grade(E,H,C)

ii) value class:NAME_school

iii) multivalued

l) Lang_level

i) description:The languages which are known and their levels.

ii) value class:LANGUAGE_LEVEL

iii) multivalued

m) Paygrade

i) value class:PAY_GRADE

ii) mandatory

n) Punishments

i) description: Summary of the personnel's criminal background.

ii) value class: PUNISHMENT

iii) multivalued

o) Awards

i) value class: AWARDS_RECEIVED

ii) multivalued

p) Courses_date

i) descriptions: Course no's and dates to which the personnel attended.

ii) value class: COURSE_DATE

iii) multivalued

q) Health_status

i) descriptions: If personnel had any major operations or important illness in the past those information can be kept with its date in this part .

ii) value class:HEALTH_STATUS

iii) multivalued

3. identifiers: GRADYEAR_DEGREE

H. INTELLIGENCE

1. description: Includes all the information about enemy forces. Alsc includes the locations of the enemy forces and the date on which the information is provided.

2. member attributes:

a) Iday

i) description: The day of intelligence is provided.

ii) value class: DATE

iii) mandatory

b) Iid

i) description: Id of the enemy installation intelligence was made on that .

ii) value class: INSTALLATION_ID

iii) mandatory

c) Itype_num

i) description: Observed weapons and the quantity of this type of weapons at that installatien. Weapon type is concatenated with the number of this type weapons observed on a particular day.

ii) value class:ITYP_NUM

iii) multivalued

iv) mandatory

d) Artiairtype_num

i) desription: Observed antiaircraft weapon systems and the quantity of this type of weapons at that installation. Weapon type is concatenated with the number of this type observed on a particular day .

ii) value class: ANTITYPE_NUM

iii) multivalued

e) Location

i) description: Locations of the anti-aircraft weapon systems. This information can be represented by geographical coordinates.

ii) value class: ANTI_AIRCRAFT_LOCATION

iii) multivalued

f) Weather

i) description: Weather status of the day on which the intelligence is provided. (especially for photo intelligence , or tactical reconnaissance missions).

ii) value class: WEATHER_STATUS

3. Identifiers: Iday + Iid + Itype_num + Antiairtype_num

I. STATUS

1. description: Includes periodic information which come from all the subparts of Air Forces . This information includes brief operational readiness in terms of equipment , material and personnel.

2. member attributes:

a) Sday

i) description: The date of the status report which is valid for that day.
ii) value class: DATE
iii) mandatory

b) Siid
i) description: Installation id which status report belongs to it.
ii) value class: INSTALLATION_ID
iii) mandatory

c) Unoprtype_num
i) description:Unoperational aircraft type and the number of them.
ii) value class: UNOPRTYPE_NUM
iii) multivalued

d) Numpers
i) description:number of the personnel which are currently operationally ready in an installation .
ii) value class: NUMBER_OF _PERSONNEL
iii) mandatory

e) Tacsta
i) description: Status of the TACAN system if there exists in that installation.
ii) value class: NAVIGATION_STATUS

f) Gcasta
i) description: Status of the GCA system if there exists in that installation.
ii) value class: NAVIGATION_STATUS

g) Rfsta
i) description:Status of the RF system if there exists in that installation.

ii) value class: NAVIGATION_STATUS

h) Ilssta

i) description: Status of the ILS system if there exists in that installation .

ii) value class: NAVIGATION_STATUS

i) Weather

i) description: Weather status of the installation area .

ii) value class: WEATHER_STATUS

iii) mandatory

3. Identifiers: Sday + Siid

J. INSTALLATION

1. description: Includes all the information on installaticns. Identification of installation,type of weapons,categcry of installation.

2. member attributes:

a)

i) description: Installation id which uniquely determines installation.

ii) value class: INSTALLATION_ID

iii) mandatory

iv) not changeable

b) Icat

i) description: Installation category.This is important in terms of assignments.Because each personnel has to be assigned to different category of installations for a certain period.

ii) value class: INSTALLATION_CATEGORY
iii) mandatory
iv) not changeable

c) Wtype_inum
i) description: The type of weapons and the quantity of them which are currently in this installation. No matter what conditions are there in. (operational or unoperational).
ii) value class: ITYPE_NUM
iii) mandatory
iv) multivalued

d) Ipers
i) description:Total number of personnel which are currently in charge of that installation.
ii) value class: NUMBER_OF_PERSONNEL
iii) mandatory

3. Identifiers: Iid

K. LOGISTICS

1. description: Includes all the information about logistics records, supply and maintenance activities and logistics management.

2. member attributes:

a) Part_no
i) description: Unique standard national part no's of all parts which are currently in Air Forces inventory.
ii) value class: PART_NO
iii) mandatory

iv) not changeable

b) Part_name

i) value class: PART_NAMES

c) Location_num

i) description: Where these parts are located and how many of them are in that location. Location and the number are concatenated.

ii) value class: LOCATION_NUMBER

iii) mandatory

iv) multivalued

d) Suppliername_address

i) description: Name and address of the supplier of a particular part are concatenated. There may be more than one supplier for some parts.

ii) value class: NAME_ADDRESS

iii) mandatory

iv) multivalued

e) Price

i) description:Price of one unit of that item.This information is needed for bookkeeping purposes. Prices may differ from one supplier to another.

ii) value class: ITEM_PRICE

iii) mandatory

iv) multivalued

f) purchase_date

i) description: The last date of the purchase for a particular item.

ii) value class: DATE

3. Identifiers: Part_no

I. MAINTENANCE

1. description: Includes all the information in terms of planning and executing the maintenance activities.

2. member attributes:

a) Eqp_no

i) description: Serial number of the equipment.
ii) value class: EQUIPMENT_NO
iii) mandatory
iv) not changeable

b) Eqpname

i) description: Short description of equipment.
ii) value class: EQUIPMENT_NAME

c) Eqpcat

i) description:Equipment category(HF,VHF,UHF,gasoline,diesel).This is a kind of additional information to equipment name.This field is optional and it can be omitted if wanted so.
ii) value class: EQP_CATEGORY

d) Arrival_date

i) description: The date on which the equipment arrived at the supply and maintenance center. This is important because of maintenance planning.
ii) value class: DATE
iii) mandatory

e) Trouble

i) description: The trouble which is reported to supply and maintenance center by the users of equipment. Equipment may not have

any trouble. It might be for periodic maintenance and calibration.

ii) value class: EQUIPMENT_TROUBLE

iii) mandatory

iv) multivalued

f) Reqparts

i) description: To keep track of the replacement part needs, we included the required parts which are needed to repair the equipment.

ii) value class: PART_NO

iii) multivalued

g) Eiid

i) description:Installation id code that the equipment belongs.This information is needed to send the equipment back to its owner.

ii) value class: INSTALLATION_ID

iii) mandatory

h) Owner

i) description: This is the owner of the equipment. This is the in base address of the owner.

ii) value class: EQP_OWNER

3. Identifiers: Eqp_no + Arrival_date + Owner

B. COURSES

1. description: Includes all courses offered by Air Force command in the past, present and future time. This information will provide planning and choosing the persons who will participate to these courses.

2. member attributes:

a) Course_no
   i) description: Each course has a unique number which is related with its topic.
   ii) value class: COURSENO
   iii) mandatory

b) Course_name
   i) value class: COURSENAME
   ii) mandatory

c) Course_location
   i) value class: COURSELOCATION
   ii) mandatory
   iii) multivalued

d) Number_of_hcurs
   i) value class: COURSE_PERIOD
   ii) mandatory

e) Start_date
   i) value class: DATE

f) Participants
   i) description: Names of the personnel whc did or will attend to this course.
   ii) value class: PARTICIPANT'S_NAME
   iii) multivalued

3. Identifiers: Ccurse_no

N. STRING TYPES

Now we should define the string types for those records. Strings are following.

1. WEAPON_TYPE

   interclass connection: Subclass of STRINGS where format is 5-alphanumeric character string. Slash and dash can be included in string.

2. FRIEND_FOE

   interclass connection: Subclass of STRINGS where format is 3-alpha character string. The values are either FRD or FOE.

3. RANGE

   interclass connection: Subclass of STRINGS where value is positive integer less than 25,000.

4. FUEL

   interclass connection: Subclass of STRINGS where the value is positive integers less than 30,000.

5. MAX_LOAD

   interclass connection: Subclass of STRINGS where value is positive integer less than 100,000.

6. NUMBER_OF_WEAPONS

   interclass connection: Subclass of STRINGS where value is in between 0..10,000.

7. DATE

   interclass connection: Subclass of STRINGS where format is ddmmyy where dd=1..31;mm=1..12 and yy=0..99.

8. ARMC_CATEGORY

interclass connection: Subclass of STRINGS where format is single letter string in A..G.

9. WARHEAD_CAT

interclass connection: Subclass of STRINGS where the value is in 1..10.

10. NAME

interclass connection: Subclass of STRINGS where format is 10-letter character string.

11. ADDRESS

interclass connection: Subclass of STRINGS where format is 15-ch/street; 10-ch/apt; 4-digit/no; 10-ch/city; 2-ch/state; 5-digit/zip;

12. GRADYEAR_DEGREE

interclass connection: Subclass of STRINGS where format is yyyy/ddd where yyyy represents the year, ddd represents the graduation degree.

13. LOCATIONS

interclass connection: Subclass of STRINGS where format is 4-digit number, which is installation id.

14. RANK

interclass connection: Subclass of STRINGS where format is 10-ch long string.

15. SPECIALTY

interclass connection: Subclass of STRINGS where format is 4-digit string.

16. MAR_STATUS

interclass connection: Subclass of STRINGS where format is single character which is in (S,M,D,W).

17. NAME_OCCUPATION

interclass connection: Subclass of STRINGS where format is concatenation of 25-Ch name/lastname string and 10-Ch long occupation string.

18. NAME_SCHOOL

interclass connection: Subclass of STRINGS where format is concatenation of 15-Ch name string and 3-Ch school string which is in (ELM,HIG,COL).

19. LANGUAGE_LEVEL

interclass connection: Subclass of STRINGS where format is 4-Ch string which are in (POOR,FAIR,GOOD,EXCL).

20. PAY_GRADE

interclass connection: Subclass of STRINGS where format is dd/l where dd is in 1..13 and l in 1..4.

21. PUNISHMENT

interclass connection: Subclass of STRINGS where format is 15-Ch string.

22. AWARDS_RECEIVED

interclass connection: Subclass of STRINGS where format is 10-Ch string.

23. COURSE_DATE

interclass connection: Subclass of STRINGS where format is 4-digit course no /ddmmyy where dd=1..31;mm=1..12;yy=0..99.

24. HEALTH_STATUS

interclass connection: Subclass of STRINGS where format is 25-Ch long string.

25. INSTALLATION_ID

interclass connection: Subclass of STRINGS where format is 4-digit string.

26. ITYPE_NUM

interclass connection: Subclass of STRINGS where format is 5-alphanumeric character string /4-digit character string.

27. ANTITYPE_NUM

interclass connection: Subclass of STRINGS where format is 5-alphanumeric character string /2-digit character string.

28. ANTIAIRCRAFT_LOCATION

interclass connection: subclass of STRINGS where the format is dd:mm;ss/d where dd=1..360;mm=1..60; ss=1..60; and d is either E or W.

29. WEATHER_STATUS

interclass connection: Subclass of STRINGS where format is 4-Ch long strings which are POOR, FAIR, GOOD.

30. UNOPRTYPE_NUM

interclass connection: Subclass of STRINGS where format is 4-alphanumeric character string / 2-digit number.

31. NUMBER_OF_PERSONNEL

interclass connection: Subclass of STRINGS where format is 4-digit strings in 1..9999.

32. NAVIGATION_STATUS

interclass connection: Subclass of STRINGS where format is 1-letter strings in (A,B,C,D).

33. INSTALLATION_CATEGORY

interclass connection: Subclass of STRINGS where format is 1-letter strings in (A,B,C,D,E).

34. PART_NO

interclass connection: Subclass of STRINGS where format is is xxx-xxxx-xxx;x is a digit.

35. PART_NAME

interclass connection: Subclass of STRINGS where format is 15-Ch long string.

36. LOCATION_NUMBER

interclass connection: Subclass of STRINGS where format is dddd/llll where dddd is installation id code d is a digit;llll is the number of this part in that location. llll is in 1..9999.

37. NAME_ADDRESS

interclass connection: Subclass of STRINGS where format is 15-Ch name,last name;10-Ch/street;4-digit/no;10-Ch/city;10-Ch/state; 5-digit/zip.

38. ITEM_PRICE

interclass connection: Subclass of STRINGS where format is ddddddddd.dd where d's are digits.

39. EQUIPMENT_NO

interclass connection: Subclass of STRINGS where format is xxx-xxxx-xxx where x's are digits.

40. EQUIPMENT_NAME

interclass connection: Subclass of STRINGS where format is 15-Ch long string.

41. EQP_CATEGORY

interclass connection: Subclass of STRINGS where format is 4-Ch long string.

42. EQUIPMENT_TROUBLE

interclass connection: Subclass of STRINGS where format is 15-alpha Character long string.

43. EQP_OWNER

interclass connection: Subclass of STRINGS where format is 15- alpha Character long string.

44. COURSENO

interclass connection: Subclass of STRINGS where format is 4-digit string.

45. COURSENAME

interclass connection: Subclass of STRINGS where format is 15-alpha Character string.

46. COURSELOCATION

interclass connection: Subclass of STRINGS where format is 10-alpha Character long string.

47. COURSE_PERIOD

interclass connection: Subclass of STRINGS where value is in 10..1000.

48. PARTICIPANT'S_NAME

interclass connection: Subclass of STRINGS where format is 10-alpha Character name/10-alpha Character last name.

V. NETWORK CONSIDERATIONS

A. INTRODUCTION

The old model a single computer serving all of the organization's computational needs is rapidly being replaced by one in which a large number of separate but interconnected computers do the job. These systems are called computer networks. Two computers are said to be interconnected if they are capable of exchanging information. By requiring that the computers be autonomous, we exclude from our definition systems in which there is a clear master/slave relation. In our view a distributed system is a special case of a network, one with a high degree of cohesiveness and transperancy.

In our information system we need a great deal of information exchange between the units. Since the system elements located geographically dispersed in order to exchange information than we need to design an information network between the units. This network will accomplish our system goals, which have been previously stated.

B. NETWORK GOALS

Air forces will have a substantial number of computers in operation, located far apart. Initially each of these computers may have worked in isolation from the other ones, at a certain time AFCOM may decide to connect them to be able to extract or correlate information about the entire system. This goal is to make all programs, data and other resources available to anyone on the network without regarding to the physical location of the resource and the user. Load sharing is another aspect of resource sharing.

In air forces environment system elements frequently request information from other ones and need some remote queries or update remote databases. All these processes need a way to accomplish these jobs. This way is a well-established information network.

A second goal is to provide high reliability by having the alternative scurces of supply. With unconnected computers if a machine goes down due to hardware failure, even thcugh there may be substantial computing capacity available elsewhere. With a network the temporary loss of single ccmputer is much less serious, because its users can often be accomadated elsewhere until service is restored. In our system a coplete loss of computing power for even a few hcurs due to some catastrophe, natural or otherwise is completely intolerable.

Another important reason for distributing computer power has to do with the relative price of computation versus communication. Until about 1970 computers were relatively expensive compared with communication facilities. The reverse is now true. In our system application data are generated at widely scattered points. Prior to 1970, it was not feasible to put the computer at each location to analyze the data because the computers were so expensive. Instead, all the data are transmitted to a central computer somewhere. Ncw the cost cf a small computer is negligible, so it becomes attractive to analyze the data at the place where it is captured, and cnly send occasional summaries back to computer center, to reduce the communication cost, which now represents a larger percentage of the total cost than it is used to. This approach results in a computer network.

Yet another goal of setting up a computer network has little tc do with networking at all. As a side effect of its other gcals, a computer network can provide a powerful communicatien medium among widely separated people. Using a

network it is easy for two or more unit which are far apart to write a report, send a message. When a change occurs the others can get the change immediately, instead of waiting for a long period of time.

Another major attraction building a large system by coupling large number of smaller machines is the expectation of a simpler software design. In our system it is possible to dedicate some (or all) of the processors to the specialized functions for example database management.

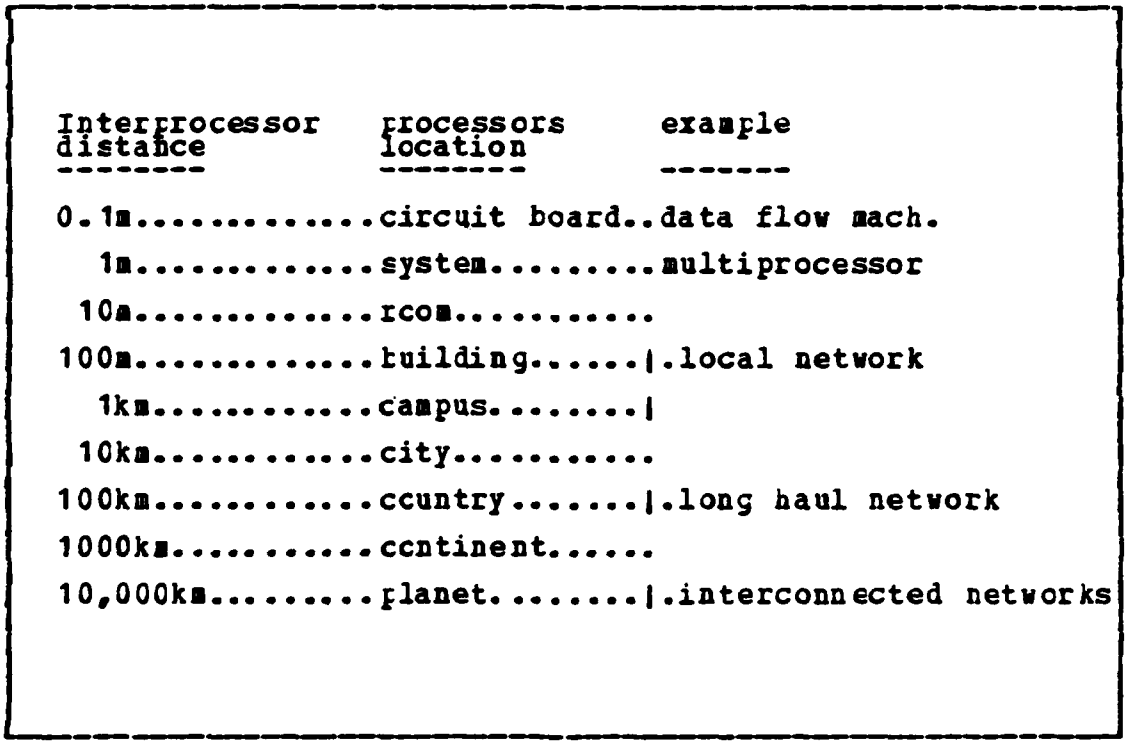


Figure 5.1 Classification Of Interconnected Processors.

By eliminating the multiprogramming, we can also eliminate much of the software complexity associated with the large mainframes.

In figure 5.1 we give a classification of multiple process systems arranged by physical size. At the top are data flow machines, highly parallel computers with many functional units all working on the same program. Next come the multiprocessers, systems that communicate by exchanging messages. Finally the connection of two or more distant network is called internetworking. In our application our system can be classified as the long haul network. Since there is only one network is established for whole system.

C. NETWORK STRUCTURE

In any network there exist a collection of machines intended for running user programs. These machines are called hosts. For instance in our system APCOM, TAF1, TAF2, and each of the bases are the hosts. The hosts are connected by the communication subnet. The job of the subnet is tc carry messages from host to host. By separating the pure communication aspects of the network from the application aspects the complete network design is greatly simplified.

In all networks the subnet consists of two basic components: switching elements and transmission lines. The switching elements might be specialized computers. We will call them IMPs (interface message processors). Transmission lines are called circuits or channels

In figure 5.2, each host is connected to one IMP. All traffic to or from the host goes via its IMP.

Broadly speaking there are two general types of designs for the communication subnet.

1. Point-To-Point channels

The network contains numerous cables on leased telephone lines, each one connecting a pair of IMPs. If two IMPs

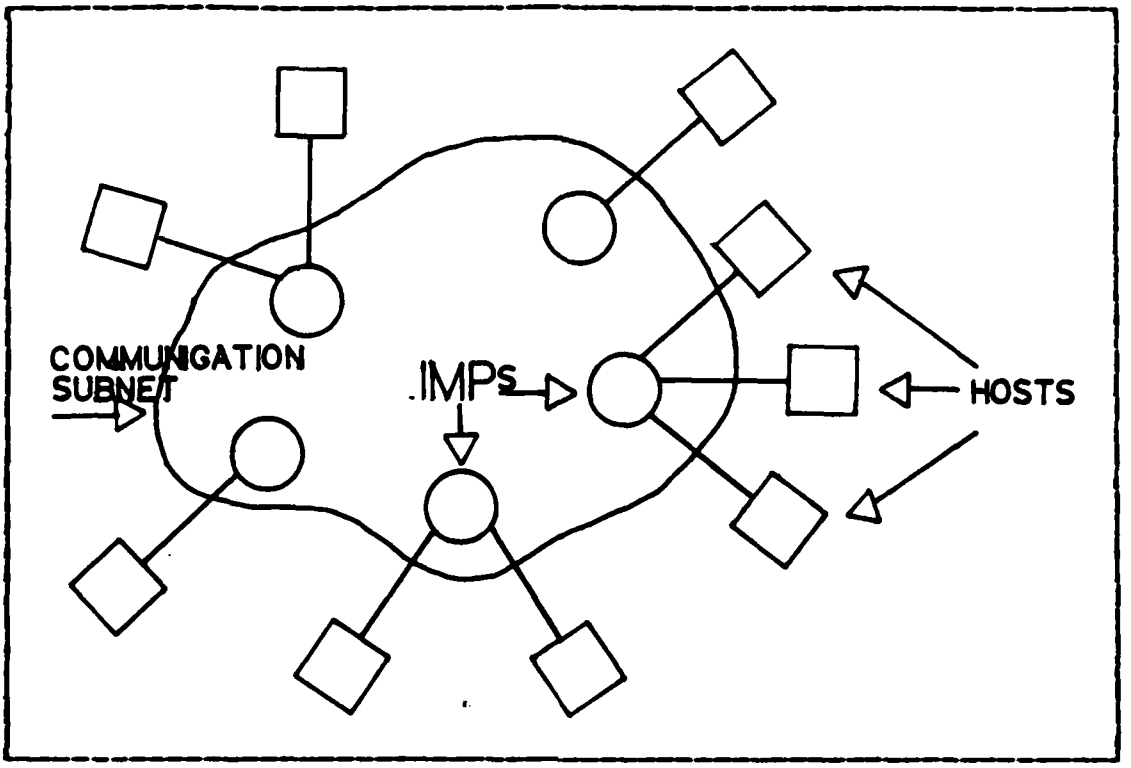


Figure 5.2 Communication Subnet.

that do not share a cable nevertheless wish to communicate they must do this indirectly via other IMPs. When a message is sent from one IMP to another IMP via one or more intermediate IMPs the message is received at each intermediate IMP, in its entirety, stored there until the required output line is free, and then forwarded. A subnet using this principle is called point-to-point or store-and-forward subnet. In our system we will use this kind of communication. Since some units need not to communicate directly with AFCOM in system hierarchy. We will take a close look at to this subject when we consider network topology.

2. Broadcast Channels

The second kind of communication architecture uses broadcasting. In this design there is a single communication channel shared by all IMPs. Inherent in broadcast system is that messages sent by any IMP are received by all other IMPs. Something in the message itself must specify for whom it is intended. After receiving a message not intended for itself, an IMP just ignores it. The most important issue in broadcast channels is the lack of privacy. Privacy is an important system aspect in our design. Thus we can not use broadcast channels in our design.

D. NETWORK TOPOLOGY

The goal of the topological design is to achieve a specified performance at a minimal cost. There are no exact solutions in this business. Trial and error plus the services of a large computer are essential. The problem is so immense that the only reasonable approach is to generate a potential network topology and then see if it obeys the connectivity and delay constraints. If not generate another one, until a feasible one is found. In our system design constraints we should first consider the interactions among the units. Since our system's nature is hierarchical and then we should first this at one side. Then, we should look at the communication issues. We have said that the broadcast communication is not suitable due to lack of its privacy. And the most important issue is the design consideration of the databases. Since we have designed the databases of the sub levels as a subschemas of the upper levels, subunits need not to communicate directly to the upmost level. First we should look at the point-to-point interconnection topology types.

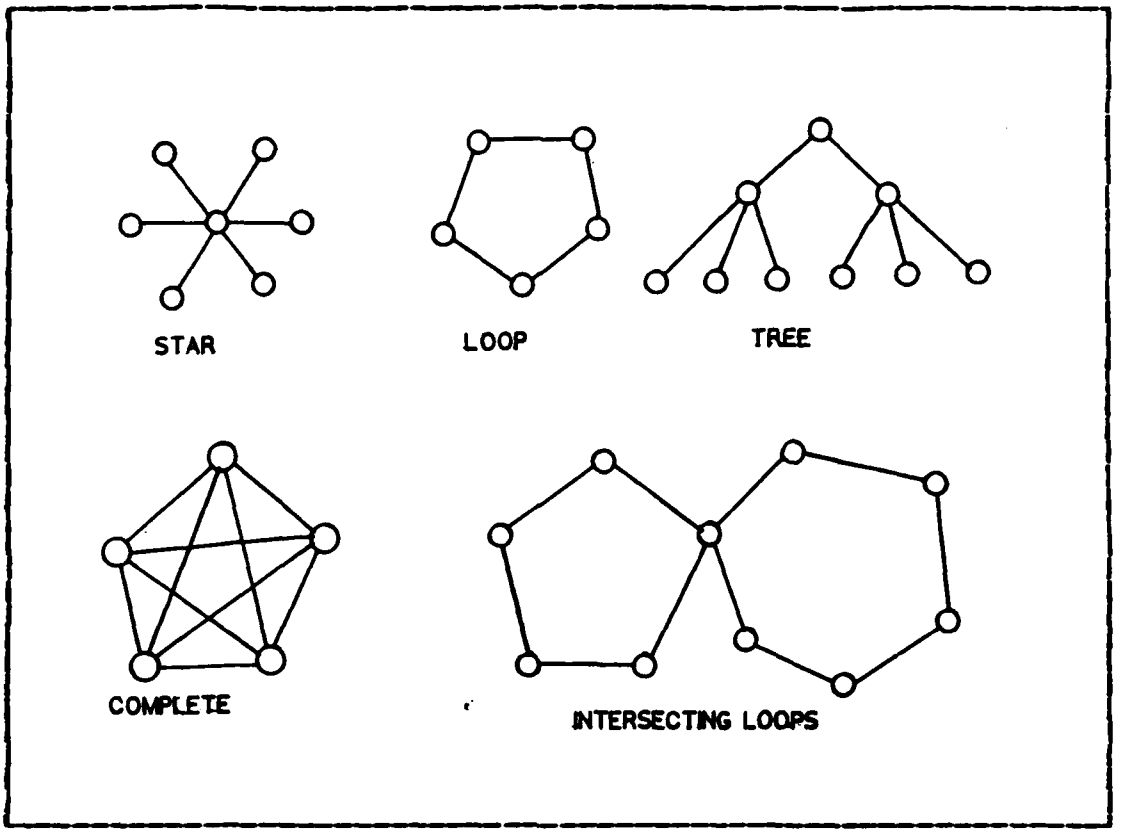


Figure 5.3 Possible Topology Types.

In Figure 5.3 we see several possible topologies. Among these topologies tree structure is the most appropriate one to satisfy our system constraints. This topology permits us to implement our network in a hierarchical way. In this structure each subunit can communicate with one-level higher unit. If this unit wants to get, or send any information from or to higher level units, it should do this via intermediate levels. For instance if a base would like to retrieve a record from AFCOM's database it should do this via its ordinate TAF. And, also if TAF1 would like to send a message to TAF2, then it should send this message via AFCOM, then AFCCM can transmit this message to TAF2. THis is very important in terms of network design, because this will

greatly simplify the design of network. Besides there is a great need to establish a connection among the supply and maintenance centers. Because the traffic is very heavy among these subunits, if we do not set up interface for these subunits then we can greatly reduce the overall system throughput. Also supply and maintenance centers pass very burstly information to each other. But the other system components will not have this big jobs to be interacted with each other subunits.

E. DESIGN ISSUES

The first design decision is the rules for data transfer. Do data only travel in one direction called simplex communication, or can they travel in either direction, but not simultaneously called half duplex communication, or can they travel in both directions at once, called full duplex communication? For our system half duplex communication is enough. Since, we have established an interface among the supply and maintenance centers, this will reduce the traffic. Thus the half duplex communication is preferable besides it is simple and efficient method.

The protocol must determine how many logical channels the connection corresponds to, and what their properties are. In our information network provides two logical channels per connection one for normal data and one for urgent data.

Error control is an important issue when the physical communication circuits are not perfect. many error detecting and correcting codes are known, but both ends of the connection must agree on which one is being used (we will look at to this issue later in this chapter). In addition the receiver, must have some way of telling the sender which messages have been correctly received and which have not.

Not all communications channels preserve the order of the messages sent on them. To deal with a possible loss of sequencing the protoccl must make explicit provision for the receiver to allow the pieces to be put back together properly.

An issue that cccurs frequently is how to keep a fast transmitter from swamping a slow receiver with data. Some kind of feedback frcm the receiver to the sender can be established related with receiver's current situation.

When there are multiple paths possible between source and destination at some point in the hierarchy, a rcuting decision must be made. Also we will look at to this issue later in this chapter. para;All these are very important issues related with the system's performance. We should consider these issues very carefully when we are designing our system.

F. NETWCRK ARCHITECTURES

To reduce the network design complexity, most networks are organized as a series of layers or levels, each cne built upcn its predecessor. The number of layers the name cf each layer and the function of each layer differ from network to network. However in all networks the purpose of each layer is to cffer certain services to the higher layers, shielding those layers from the details of how the cffered services are actually implemented. Layer n on one machine carries on a conversation with layer n on another machine. The layers and convention used in this conversation are collectively knowm as the layer n protocol, as illustrated in Figure 5.4 for a seven layer network.

In reality no data are directly transformed from layer n on one machine to layer n on another machine (except in the lowest layer). Instead each layer passes data and

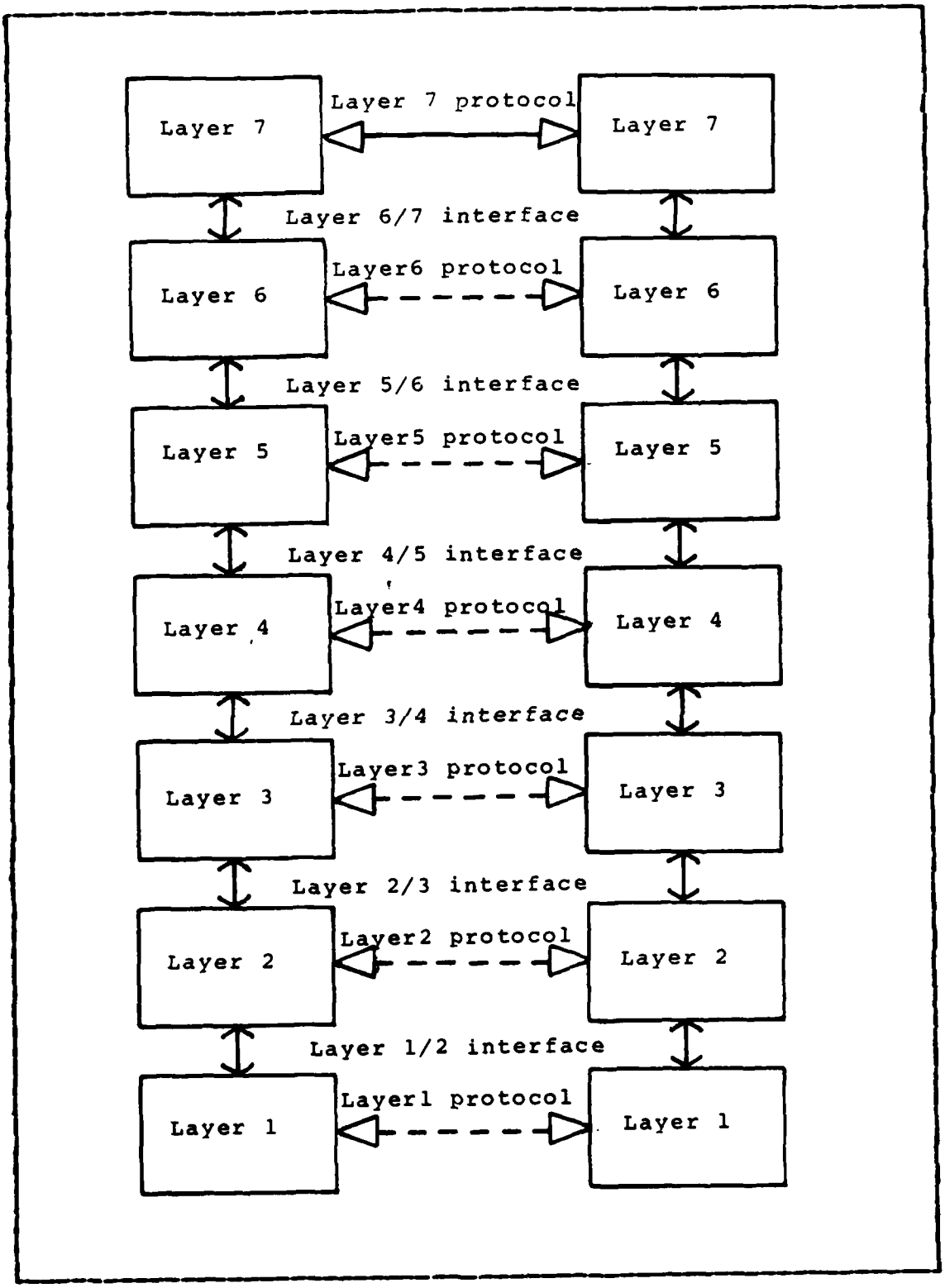


Figure 5.4 ISO Seven Layer Architecture.

control information to the layer immediately below it, until the lowest layer is reached. At the lowest layer there is physical communication with the other machine as opposed to the virtual communication used by the higher levels. In Figure 5.4, virtual communication is shown in by dotted lines and the physical communication is shown by the solid lines.

Between each pair of adjacent layers there is an interface. The interface defines which primitive operations and services the lower layer offers to the upper one. When we decide how many layers to include a network and what each one should do one of the most important considerations is having cleanly defined interfaces between the layers. Having cleanly defined interfaces in turn requires that each layer perform a specific collection of well understood functions.

The set of layers and protocols is called network architecture. the specification of the architecture must contain enough information to allow an implementer to write the program for each layer so that the program for each layer so that the program will correctly obey the appropriate protocol.

We will only consider the ISO (International standards organization) seven layer model. We will not look at the other architectures like ARPANET, IBM's SNA, and DEC's DECNET. Now we should look at the layers of ISO model.

1. The Physical layer

The physical layer is concerned with transmitting raw bits over a communication channel. Typical questions here are how many volts should be used to represent a 1, and how many microseconds a bit occupies, whether transmission may proceed simultaneously in both directions, how the initial connection is established and how it is torn down when both sides are finished, how many pins the network

connector has and what each pin is used for. We will use the leased telephone lines as physical connections among the hosts.

2. The Data Link Layer

The task of the data link layer is to take a raw transmission facility and transform it into a line that appears free of transmission errors to the network layer. It accomplishes this task done by breaking the input data up into data frames, transmitting the frames sequentially and processing the acknowledgement frames sent back by the receiver. Since layer 1 merely accepts and transmits a stream of bits without any regard to meaning or structure, it is up to the data link layer to create and recognize frame bcundaries. This can be accomplished by attaching special bit patterns to the beginning and the end of the frame.

A noise burst on the line can destroy a frame completely. In this case the layer 2 software on the scurce machine must retransmit the frame. Another issue that arises at layer 2 is how to keep a fast transmitter from drcwning a slow receiver in data. A mechanism must be employed to let the transmitter know how much buffer space the receiver at the moment.

3. The Network Layer

the network layer ccntrols the operation of the subnet. Amcng other things, it determines the chief characteristics of the the IMP host interface and how packets the units cf information exchanged in layer 3 are routed with in the subnet. A major design issue here is the division of labor between the IMPs and Hosts, in particular whc should ensure that all packets are correctly received at their destinations, and in the proper order. What this layer

software does basically is accept messages from the scurce host, convert them to packets and see to it that the packets get directed toward the destination. A key design issue is how the route is determined. We will consider this issue later in this chapter.

4. The Transport Layer

The basic function of the transport layer also known host-to-hcst layer is to accept data from the session layer split it up into smaller units, if need be pass these to the network layer, and ensure that the pieces all arrive at the cther hand.

the transport layer also determines what type of service to provide the session layer, and ultimately the users cf network. The most popular type of transport connection is error free (virtual) point-to-point channel that delivers message in the order in which they were sent.

Although the network architecture specifies nothing about the implementation, it is worth pointing out that the transpcrt layer is often implemented by a part of the host cperating system, which we will call a transport station. In contrast the network layer is typically implemented in the host by an input/output driver. The data link and physical layers are normally implemented in hardware.

5. The Session Layer

The session layer is the user's interface into the network. It is with this layer that the user must negotiate to establish a ccnnection with a process on ancther machine. Once the connection has been established the session layer can manage the dialog in an orderly manner if the user has requested that service. A connection between users is usually called session. A a session might be used to allcw a user to log into a remote time sharing system or

to transfer a file between two machines. To establish a session the user must provide the remote address he wants to connect to. Session addresses are intended for use by users or their programs whereas transport addresses are intended for the use by transport stations, so the session layer must be able to convert a session address to its transport address, to request that a transport connection be set up.

6. The Presentation Layer

the presentation layer performs functions that are requested sufficiently often to warrant finding a general solution for them, rather than letting each user solve the problems. These functions can often be performed by library routines called by the user.

A typical example of a transformation service that can be performed here is text compression. The presentation layer could be designed to accept ASCII strings as input and produce compressed bit patterns as output.

This layer can also perform other transformations in addition to message compression. Encryption is one possibility. Conversion between character codes, line and screen length, end of line convention, scroll versus page mode, character sets and cursor addressing are but a few of the many problems. The representation layer attempts to alleviate these problems.

7. The Application Layer

The content of the application layer is up to the individual user. When two user programs on different machines communicate they alone determine the set of allowed messages and the action taken upon receipt of each. Nevertheless there are many issues that occur here that are quiet general.

It is generally inadvisable to create our own network software. The job inevitably seems to take much more programming time than anticipated. A do-it- yourself approach may be seen reasonable for simple point-to-point or dial telephone lines between processors. But our system can grow rapidly in a few years. Thus this kind of more complex forms of transport network will need complex software. These are sufficiently complex that a do-it yourself approach is not advisable.

A full-function architecture for distributed processing from a given manufacturer should do everything listed in this section. So we will look at only designing issues rather than designing our network software. [Ref. 4]

- MESSAGES, TRANSACTIONS AND BATCHES

A message is a group of bits sent as a single communication between two machines. It is possible that for technical reasons, a message may be divided into separate blocks for transmission (sometimes called packets). If that happens, the message will be reassembled before it is given to the process which uses it.

A transaction consists of the data which form a basic unit of work for the application in question. To complete one transaction several messages may have to go back and forth.

A group of transactions is sometimes sent together in one transmission, for example from a peripheral processor used for data entry. This is referred as a batch. It may sent as multiple transmitted blocks, depending on what block size gives the most efficient transmission.

8. Line Control

Teleprocessing needs a line control procedure to perform the following functions:

1. Indicate start and end of a message.
2. send an address with the message so that it goes to the requisite device.
3. Detect transmission errors.
4. Initiate retransmission of messages in error, or messages failed to arrive.
5. Maintain line discipline to ensure that two devices do not transmit at once, and garble each other's message.

The designer selected processors to be interlinked must ensure that they use exactly compatible line-control procedures. Some line control procedures such as start-stop control with with the CCITT alphabet No.5(ASCII) [Ref. 5] IBM's SDLC [Ref. 6] is a subset of HDLC. Most other manufacturers have their own version of HDLC.

9. Code Selection

To communicate the processors selected must be able to use not only the same line control procedure but also the same character set, possibly the ASCII code or CCITT Alphabet No.5. If we buy IBM's products then we have to use EBCDIC.

10. Message Header

A variety of control information may be carried in the header of each message. The system designer who has elected to use conventional teleprocessing rather than a network architecture must determine what information is

needed in the message headers. The following are some of the types of header information that might be necessary:

a. TYPE OF MESSAGE

Often a machine will receive more than one type of message. The header should be indicate the type possibly one Byte.

b. DESTINATION ADDRESS

In our system message can travel over more than one physical links. A destination address is needed to tell the concentrater or switch where to route the message.

c. SOURCE ADDRESS

The receiving processor needs to know where a message has come from. If it has traveled more than one link via a concentrater or switch, it must carry the address of its source. this address is placed in any response messages.

d. MESSAGE SERIAL NUMBER

The messages may be given a sequential number. This may be used for the following purposes:

1. For that no message is lost.
2. For associating a response with the message triggering that response.
3. If a long message is split into block or packets for reassembling that message.
4. For identifying a message for audit purposes.

e. TRANSACTION SERIAL NUMBER

When several messages are involved in one transaction, the transaction may be given a serial number rather

than irdividual messages. This enables the transactien to be traced and referred for audit, testing or retrieval purposes. If it is possible for one message within a transaction to be missing the message within a transaction may be numbered.

f. CHAINING INDICATOR

If a trarsaction can be split into messages a chaining indicator may indicate that more messages will follow in that transaction. Two bits are sometimes used in the chaining indicator, having the following meanings:

1. First message in transaction.
2. Middle message.
3. Last message in transaction.
4. Only message in transaction.

g. MESSAGE REASSEMBLY INDICATOR

If a long message is chopped up into blocks or packets, sufficient information must be included to reassemble the message and detect whether any packet is lost. This is usually dcne with a chaining indicator and sequential number.

h. RESPONSE INDICATOR

Sometimes when a processor sends a message, it requires a response; It may be programmed to wait until it receives a response. In this case it should send an indication in the message saying that it is waiting. A 2-bit indicator is sometimes used having the following meaning:

1. No response expected.
2. A response must be sent to this message.
3. A response must be sent to this chain of messages after the last message in the chain is indicated.

4. A message may or may not be sent.

i. TIME AND DATA STAMP

We need tc know the time and the date of the messages that they were sent. This is used for reference and audit purpcses. Thus the messages should be marked to indicate the time and date they were sent.

j. BATCH CONTROLS

In our system we usually transmit a batch of transactions rather than individual transactions. Demarcaticn controls are needed in the batch to indicate the start and end of individual records or transactions and the end of batch. At the end of the batch a control record should be sent so that the receiving machine can check that the transactions received are complete and accurate. This control record may ccntain a count of the transactions and a hash total.

The receiving machine adds up this total and ensures that it agrees with the total written by the transmitting machine in the ccntrol record.

11. Ccntrol Messages

To regulate the transfer of information and deal with exception conditions, certain messages are needed which do not carry application data but which are control signals. The following types cf control messages are used:

a. READY TO BEGIN

One machine signals to another that it is ready to receive transmissicn.

b. NOT READY

A machine indicates that it is not at the moment that it is ready to receive.

c. REQUEST PERMISSION TO TRANSMIT

One machine contacts another, requesting permission to transmit.

d. REJECT

One machine rejects the transmission from another, indicating the reason. The rejection will not normally be due to a transmission error because these are detected and dealt with by the line control procedure. It will be a validity error or failure message of some type.

e. RETRANSMISSION REQUEST

The receiving machine request that a message with a given number be resent; or possibly all messages after a given number. This may be necessitated by a machine or line failure of some type.

f. INTERRUPT

One processor may send an urgent message to interrupt another processor.

g. INITIATE SESSION

Various types of message may be interchanged to initiate a session between two machines or to check that the messages have permission and have the resources to communicate. Security checks may be necessary before the session can commence.

12. Compaction

If transmission efficiency is a major concern, data may be compressed before transmission. This can reduce the number of bits by one third or one half [Ref. 7] The compressed message will be restored to its original form at the other end of the link before use. The compression and expansion may be done by software in the using computers. Like cryptography, it may also be done by intelligent hardware external to the computer [Ref. 1]

G. ROUTING

The routing algorithm is that part of the network layer software, responsible for deciding which output line an incoming packet should transmitted on.

Regardless of whether routes are chosen independently for each packet or just at the start of new sessions, there are certain properties that are desirable in a routing algorithm: correctness, simplicity, robustness, stability, fairness, and optimality.

Once a network comes on the air, it may be expected to run continuously for years without system-wide failures. During that time there will be hardware and software failures of all kinds. Hosts, IMPs and lines will go up and down repeatedly and the topology will change many times. The routing algorithm must be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network to be rebooted every time some IMP crashes.

As a compromise many networks attempt to minimize the number of the hops tends to improve the delay and also reduce the amount of bandwidth consumed, which tends to improve the throughput as well.

Routing algorithms can be classed into two classes: nonadaptive and adaptive. Nonadaptive algorithms do not base their routing decisions and measurements or estimates of the current topology and traffic whereas the adaptive ones do. Adaptive algorithms can further be subdivided into centralized, isolated, and distributed [Ref. 8]

We will not here look at all routing algorithms. We will only lock at the static routing which meets all of our design specifications. Since our network does not have too many nodes and static routing is the simplest one. Besides if we can make a good estimate about alternate routes, static routing can be the most efficient one for our system. Fortunately we have small number of nodes which permit us to make a good estimation on alternate routes.

1. Static Routing

Static or directory routing is a simple algorithm and one of the most widely used. Each IMP maintains a table with one row for each destination IMP. A row gives the best, second best, third best and so on, outgoing line for that destination, together with a relative weight. Before forwarding a packet, an IMP generates a random number and then chooses among the alternatives, using the weights as probabilities. The tables are normally worked out by the network operators, loaded into the IMPs before the network is brought up, and not changed there after.

Actually in our network we don't have a big routing problem. Since the hosts do not have more than one connection at any level. Thus if TAF1 would like to send a packet to TAF2 there is only one route via AFCOM. It is obvious that we don't need to determine which route is better than the others. In future if we make a lateral connections in this network then we will have a serious routing problem. In that case we should reconsider these routing issues.

The only problem with static routing is the determination of weights for each route. This job should not be ad hoc. A better although more complicated way to choose the routes explicitly minimize the mean packet time [Ref. 9]

Static directory routing can give good performance if the topology and the traffic do not change much. It also makes good use of existing bandwidth by using alternative routes.

H. TRANSMISSION ERRORS

Transmission errors are the fact of life. Network designers have developed two basic strategies for dealing with errors. One way is to include enough redundant information along with each block of data sent to enable to the receiver to deduce what the transmitted character must have been. The other way is only to include enough redundancy to allow the receiver to deduce that an error occured but not which error, and have it request a transmission. The first strategy is error correcting codes, and second uses error detecting codes.

1. Error Correcting Codes

A message consists of m message bits, and r check bits. Let the total length be n. An n bit unit containing data and check bits is often referred to as an n bit codeword. The error detecting and correcting properties of a code depend on its Hamming distance [Ref. 10] The number of bit positions in which two codewords differ is called the Hamming distance d apart, it will require d single-bit errors to convert one into the other. To detect d errors, we need a distance d+1 codeword because with such a code there is no way that d single-bit error can change a valid codeword into another valid codeword. When the receiver sees an

invalid codeword, it can tell that a transmission error has occured. Similarly to correct d errors we need a 2d+1 code because that way the legal codewords are so far apart that even with d changes, the original codeword is still closer than any other codeword, so it can be uniquely determined.

The bits of the codeword are numbered consecutively starting with bit 1 at the left end. The bits that are power of two are checkbits. The rest are filled up with the m data bits. Each check bit forces the parity of some collection of bits, including itself, to be even or odd. A message bit may be included in several parity computations. To see which check bits the data bit in position k, contributes the, rewrites k as the sum of powers of 2. For example, 29=1+4+8+16. A bit is checked by just those check bits occuring in its expansion. When a codeword arrives, the receiver initializes a counter to zero. It then examines each check bits, k to see if it has the correct parity. If not it adds k to the counter. If the counter is zero after all the check bits have been examined, the codeword is accepted as valid. If the counter is nonzero it contains the number of incorrect bit.

Hamming codes can only correct single errors. However there is a trick that can be used to permit Hamming codes to correct burst errors. A sequence of k consecutive codewords are arranged as a matrix, one codeword per row. Normally the data would be transmitted one codeword at a time, form left to right. To correct burst errors the data should be transmitted one column at a time starting with the leftmost column. When all k bits have been sent, the second column is send and so on. When the message arrives at the other side the matrix is reconstructed, one column at a time. If a burst error of length k occurs, 1 bit in each of the k words will have been affected, but the Hamming code can correct one error per codeword, so the entire block can be restored.

2. Error-Detecting Codes

Error correcting codes are sometimes used for data transmission, for example when the channel is simplex, so retransmission can nct be requested, but most often error detection fcllowed by retransmission is preferred because it is more efficient. In practice a method is in widespread use: the polynomial code(also known as cyclic redundancy code or CRC). Polyncmial codes are based upon treating the bit strings as representations of polynomials with coefficients of 0 and 1 only. A k-bit message is regarded as the coefficient for a polynomial with k terms. Such a polynomial is said to be of degree k-1. For example, 11000 has five bits and thus represents a five term polynomial with coefficients 1, 1, 0, 0, and 0. Polynomial arithmetic is done module 2, according to the rules of algebraic field theory. There are no carries for addition and no borrows for subtraction. Both subtraction and addition identical to exclusive or.

When the polynomial code is used, the sender and receiver must agree upon a generator polynomial, G(x) in advance. Both the high and low order bits of the generator must be 1. To compute the checksum for some messages with m bits, ccrresponding to the polynomial M(x)the message must be lcnger than the pclynomial. The basic idea is to append a checksum to the end cf the message in such a way that the polyncmial represented by the checksummed message is divisible by G(x). If there is a remainder, there has been a transmission error.

The algorithm for computing the checksum is as follows:

1. Let r be the degree of G(x). Append r zero bits to the lcw order end of the message, so it now contains m+r bits.

2. Divide the string corresponding to G(x) into the bit string corresponding to r zero bits appended message.
3. Subtract the remainder (which is always r or fewer bits) from the bit string corresponding to r zero bits appended message, using modulo 2 subtraction. The result is the checksummed message to be transmitted.

I. CRYPTOLOGY

Until the advent of computers, one of the main constraints on cryptography has been the ability of the code clerk to perform the necessary transformations, often a battlefield with little equipment. An additional constraint has been difficulty in switching over quickly one cryptographic method to another one since this entails retaining a large number of people. The danger of a code clerk being captured by the enemy has made it essential to be able to change the cryptographic method instantly. These conflicting requirements have given rise to the model of Figure 5.5 .

The messages to be encrypted known as the plaintext, are transformed by a function that is parameterized by a key. The output of the encryption process known as the ciphertext, is then transmitted by a messenger or radio. We assume that the enemy or intruder hears and accurately copies down the complete ciphertext. Sometimes the intruder does not only listen to the communication channel but can also record messages and play back them later, inject his own messages, or modify legitimate messages before they get to the receiver.

The key consists of a short strings of characters that selects one of many potential encryptions. In contrast the

First and the oldest cipher known is the CEASER cipher. In this method each letter is substituted with a 3-letter shifted one. For instance a becomes D, b becomes E and so on. A slight generalization of this cipher allows the ciphertext alphabet tc be shifted by k letters, instead of always 3. In this case k becomes a key to the general method of circularly shifted alphabets.

The next improvement is to have each of the symbols in the plaintext, say 26 letters for simplicity, each map onto scme other letter. For example, first row is a plaintext and the seccnd row is ciphertext.

a b c d e f g h i j k l m n o p q r s t v y w x y z
s t l y h o p w b x u c z a f c i d j v e m k n q r

This general system is called monoalphabetic substitution, with the key being the 26-letter string corresponding to the full alphabet.

Given a surprisingly small amount of ciphertext, the cipher can be broken easily. The basic attacks take advantage of statistical properties of natural languages. In english, for example, e is the most common letter, followed by t, a, o, n, i. The most common letter combinaticns, or digrams, are: th, in, er, and an. The most common three letter ccmbinations, cr trigrams are: the, and, ion and end.

A cryptanalyst trying to break a monoalphabetic cipher wculd start out by counting the relative frequencies of all letters in the ciphertext. Then he might tentatively assign the most common one to e and the next common cne to t. Then by making guesses at common letters digrams, and trigrams, cryptanalyst builds up a tentative plaintext letter by letter.

To make the cryptanalyst's job more difficult, it is necessary to smooth out the frequencies of the ciphertext,

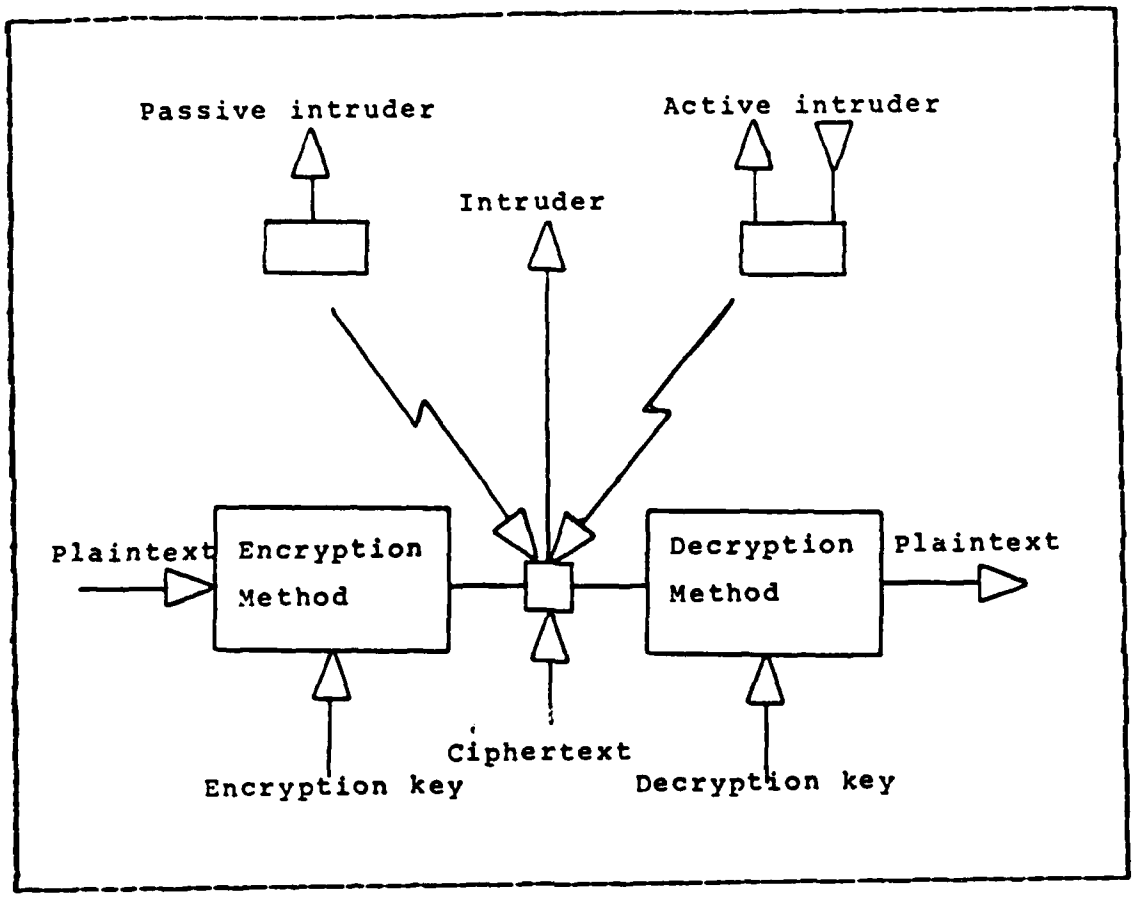


Figure 5.5 Cryptographic Method.

general method, which may be every few years, the keys can be changed as often as required. Thus our basic model is a stable and publicly known general method parameterized by a secret and easily changed key.

Encryption methods historically divided up into two categories: substitution ciphers and transposition ciphers. Now we will look at each of them.

1. Substitution Ciphers

In a substitution cipher each letter or group of letters is replaced by another letter or group of letters to disguise it.

so the letters representing e, t, do not stand out so clearly. One way of achieving this goal is to introduce multiple cipher alphabets, to be used in rotation, giving what is known as a polyalphabetic cipher. As an example consider the VIGENERE cipher. It consists of a square matrix containing 26 CEASER alphabet. The first row called row A, is A B C ... X Y Z. The next row is called row B, is B C D E F ... Y Z A. The last row, called row Z, is Z A B C D ... W X Y.

Like the moncalphabetic cipher, this cipher also has a key, but instead of being a string of 26 distinct characters, the key is usually a short, easy-to-remember word or phrase, such as EXTERRESTRIAL.

E X T E R R E S T R I A L E X T E R R E S T R
t a c t i c a l f o r c e s w i l l s t a r t

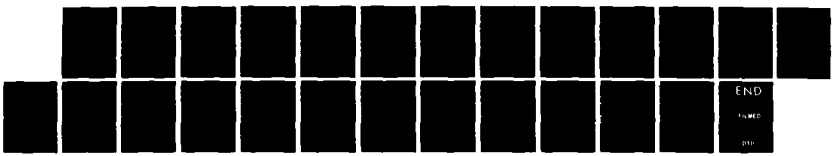
The key letter above each plaintext letter tells which row to use for encryption. The t is encrypted using the CEASER alphabet row E, then a is encrypted using the CEASER alphabet row X, and so on.

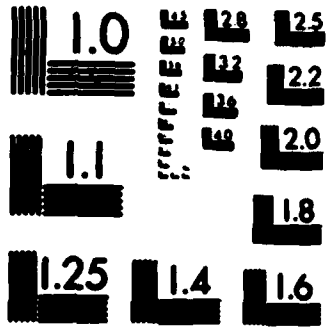
A more powerful polyalphabetic cipher can be constructed by using arbitrary monoalphabetic ciphers for the rows instead of restricted them to CEASER ciphers. The only problem with this scheme is that the 26x26 square table then becomes part of the key and must also be memorized or written down.

The next step up in the complexity for the cryptographer is to use a longer key than the plaintext. In fact, constructing an unbreakable cipher is easy. First choose a random bit string as a key. Then convert the plaintext into a bit string for example using ASCII representation. Finally compute the EX-OR of these two strings, bit by bit. The resulting ciphertext cannot be broken, because every possible plaintext is an equally probable candidate. The

AD-A150 611 GENERAL DESIGN CONSIDERATIONS OF AN AIR FORCE INFORMATION SYSTEM(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA E AYTACER JUN 84 2/2

UNCLASSIFIED F/G 9/4 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

ciphertext gives the cryptanalyst no information at all. In a sufficiently large sample of ciphertext, each letter will occur equally often, as will every digram and every trigram.

This method is known as the one time key, has a number of practical disadvantages, unfortunately. To start with, the key can not be memorized, so both sender and receiver must carry a written copy with them. Written keys are undesirable. Additionally the total amount of data can be transmitted is limited by the amount of key available. If the spy strikes it rich and discovers a wealth of data, he may find himself unable to transmit it back to headquarters because the key has been used up. Another problem is the sensitivity of the method to lost messages, or messages that arrive in the wrong order. If the sender and the receiver get out of synchronization as to where in the key they are, they are in trouble.

Substitution ciphers need not always work one letter (or bit) at a time. For example PORTA's cipher uses 26x26 table, like the VIGENERE cipher. The plaintext is encoded two characters at a time. The first character indicates a row, the second a column. The number of letter paid found at the intersection is the encrypted value. If 26 different tables are prepared, trigrams can be encrypted as units by using the first letter of each trigram to select a table.

2. Codes

As the units encrypted become longer and longer, the cipher begins to resemble a code. The main difference between a cipher and a code is that the former encrypts a fixed-size unit of plaintext with each operation, whereas the later encrypts a single variable-length linguistic unit, typically a single word or phrase.

Codes have the disadvantage of requiring large blocks, that can not be replaced as easily as the key to a

cipher. However they have the advantage of being harder to break than ciphers. Codes and ciphers can be combined to make the cryptanalyst's life less pleasant. For example, encoding a message might yield a five-digit numbers. These numbers could be concatenated to form a digit sequence, that could then be encrypted using a polyalphabetic cipher. Enciphering a coded message is called superenciphersent. Superenciphered codes are harder to break.

3. Transposition Ciphers

Substitution ciphers and codes preserve the order of the plaintext but disguise them. Transposition ciphers, in contrast, reorder the letters but do not disguise them. Figure 5.6 depicts a common transposition cipher, the columnar transposition. The cipher is keyed by a word of phrase not containing any repeated letters. In this example TROUBLE is the key. The purpose of the key is to number the columns, column 1 being under the key letter closest to the start of alphabet, and so on. The plaintext is written horizontally, as a series of rows. The ciphertext is read out by columns, starting with the column whose key is the lowest.

To break a transposition cipher, the cryptanalyst must first be aware that he is dealing with a transposition cipher. By looking at the frequency of e, t, a, o, i, n, it is easy to see if they fit the normal pattern for plaintext. If so the cipher is clearly the transposition cipher, because in such a cipher every letter represents itself.

Some transposition ciphers accept a fixed-length block of input and produce a fixed-length block of output. These ciphers can be completely described by just giving a list telling the order in which the characters are to be output.

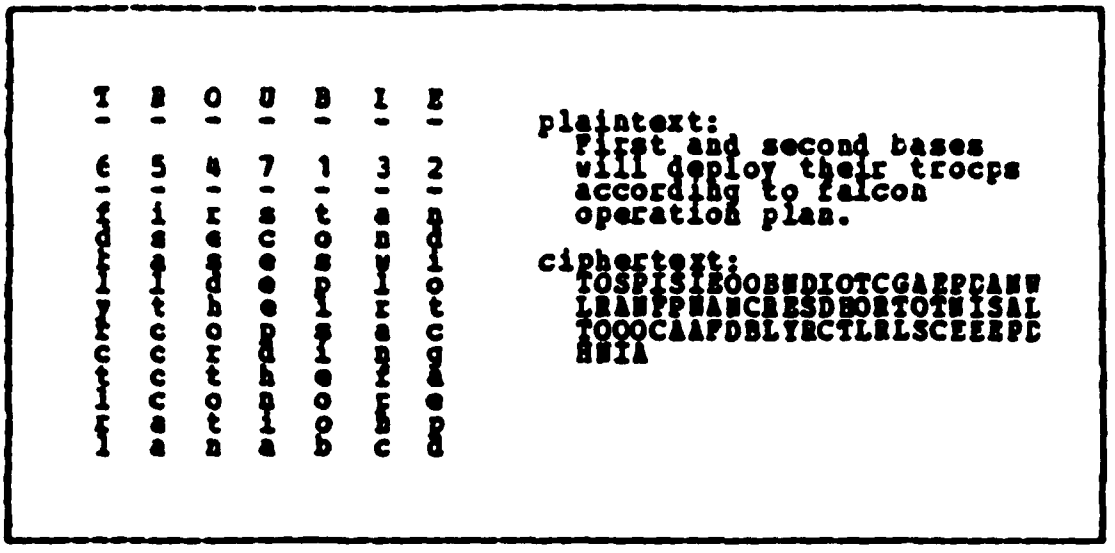


Figure 5.6 Transposition Cipher.

4. The Data Encryption Standard

While describing the various classical cryptographic schemes, we have tried to make it clear how computers can be used as powerful tools by the cryptanalyst, both for collecting frequency statistics and for trying out large numbers of tentative solutions. We will think about making the encryption process so complicated that not even a computer can break it.

Although modern cryptography uses the same basic ideas as traditional cryptography, transition and substitution, its emphasis is different. Traditionally cryptographs used simple algorithms and relied on long keys for their security nowadays reverse is true: the object is to make the encryption algorithm so complex. And involuted that even if the cryptanalyst acquires vast sounds of enciphered text of his own choosing, he will not be able to make any sense of it at all.

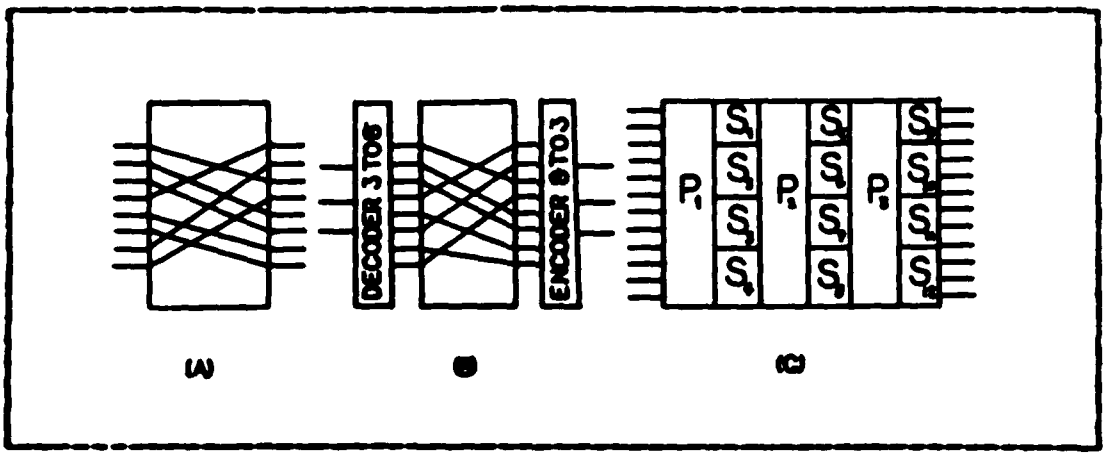


Figure 5.7 A: P-Box, B: S-Box, C: Cascade Ciphers.

Transposition and substitution can be done with simple circuits. Figure 5.7 A, shows a device known as P-box, used to effect a transposition on an 8-bit input. If the 8 bits are designated from top to bottom as 01234567, then the output of this particular p-box is 36071245. By appropriate internal wiring a p-box can be made to perform any transposition. Substitutions are performed by what are called s-boxes, as shown in Figure 5.7 B. In this example a 3-bit plaintext is entered and a 3-bit ciphertext is output. The 3-bit input selects one of the lines exiting from the first stage and sets it to 1; all the other lines are 0. The second stage is a p-box. The third stage encodes the selected input line in binary again. With the wiring shown, if the eight octal numbers 01234567 were input one after another, the output sequence would be 24506713. Again by appropriate wiring of the p-box, any substitution can be accomplished.

The real power of those basic elements only becomes when we cascade a whole series of ciphers, as shown in figure 5.7 C. In this example, 12 input lines are transported by the first stage. Theoretically, it would be

possible to have the second stage be an S-box, that mapped a 12-bit number onto another 12-bit number. However, such a device would need 2**12 = 4096 crossed wires in its middle stage. Instead the input is broken up into four groups of 3 bits, each of which is substituted independently of the others. Although this method is less general, it is still powerful. By including a sufficiently large number of stages in the product cipher, the output can be made to a nonlinear function of the input [Ref. 4]

We have talked about too much on cryptology so far, because of our system needs very tight security measures. It is obvious, that the cryptology is not only the way to prevent malicious act, to gain access to secret information. We should also consider the overall system security and privacy. But we will not consider them in this thesis, since they are very wide subjects and depend on the particular application.

VI. COST CONSIDERATIONS

A. PRELIMINARY CONSIDERATIONS

Minicomputers have historically lacked the power of mainframes. Their lower price, however, made the systems ideal for dedicated tasks. Technological advances, quickly elevated minicomputers to the class of standalone, general-purpose systems, and manufacturers began developing high-end miniccaputers with 32-Bit architectures to challange mainframe markets.

1. Defining Needs

The first task in purchasing a minicomputer is to develop a definition of the use for the new system. By detailing the environment and utilization, an organization can quickly determine whether or not a full-powered minicomputer is needed. Our system will have to support approximately 24 interactive users and provide such services as electronic mail, data base management, and advanced accounting functions. In addition the system will be required to perform batch updates to the mainframe several times a day. Also, we strongly desire that the minicomputers have several communication capabilities in terms of networking.

Examining the capabilities of alternate types of systems will lead this organization to the conclusion that minicomputers represent the best option. The single-user 8-and16- Bit microcomputer systems would obviously not satisfy our system requirements. High-end microcomputers could certainly provide necessary throughput, but this quickly degrades as additional users are added. On the other

and of the market, it would be hard to justify a mainframe for 24 users.

It is this type of dedicated application that many experts predict the minicomputer will continue to fulfill. Minicomputers will also be required for other such applications as controlling clusters of single-user microcomputers, dedicated communication switching and managing various network functions.

2. Throughput

Once the application is defined, one should estimate the required throughput. Throughput can best be defined as the system's ability to accept, process and output the results of transactions. The number of transactions, the number of users, and the amount of data that is involved will provide a relative throughput requirement.

Actual throughput is determined by a combination of processing power, generally expressed in millions of instructions per second (MIPS), bus transfer rates and memory cycle time. Such factors as the interrupt system, operating software and the availability of cache memory also effects throughput, and any one of these elements is capable of distorting the overall picture. A system featuring a high level of processing power, for example may provide unacceptable response because of operating system bottlenecks encountered when several users access it simultaneously.

The ideal system must offer the necessary support for immediate needs. It is also important that the system be able to support future expanded requirements. This support may reside in the system itself, or in the form of available upgrades.

3. User Requirements

Some systems support multiple users through multiplexed RS-232 interfaces. Others may implement direct connections for each terminal using clustered controllers. Once again, the architecture will affect the throughput and response time of the system.

One should additionally understand the vendor's method of supporting multiple users. Potential buyers should determine how interrupts are assigned and activated. A system based on priority interrupts may not provide the response that low priority users require if a large percentage of the higher priority users commonly run CPU intensive programs. On the other hand, a time-slice or polling structure may frustrate users running long or complicated tasks.

Most minicomputer vendors configure their systems with a limited number of high-speed ports and direct memory access channels. These ports can speed transactions and, depending on the application, may be the key elements of a system's success. For example an application requiring intensive updates to a disk resident data base would benefit from such a high-speed channel.

B. THE HARDWARE CONFIGURATION

In determining the application, it is important to quantify the amount of data required. Purchasers should examine such factors as the type and frequency of hard-copy reports, the amount of data that will be stored and the method used to backup that data. The peripherals supporting these functions can seriously degrade the overall performance of the system if chosen improperly.

Printers should be selected to provide both the speed and print quality that an application requires.

The amount and type of mass storage supported on a system should also be considered. All systems support disk drives, but they vary significantly in their capacities. The technology used in the disk drive can also be important.

Another element that should be examined is the method of copying data for archival storage. Although reel-to-reel magnetic tape is the most commonly used device for this operation, an application may dictate the implementation of cartridge disks or streaming tape drives. The recording speed of backup device and the ability to store the media off-line are primary considerations in this area.

1. Communications

One advantage that minicomputers presently have over microcomputers is communications support. Implementing a minicomputer in a distributed network requires a data communications link and protocol. It is, therefore, very important that the host and minicomputer be able to talk with with one another.

A common solution to this problem is through remote job entry. Most vendors, in addition to RJE, offer emulation software as well as IBM SNA and X.25 protocol support. Several vendors have also developed proprietary networking protocols. The type of link that will be established between the mainframe and the minicomputers will depend on the application. Remote job entry may be the most effective approach for a remote system which only updates a centralized data base at the end of each day. A system that interacts with a host several times a day, however, will need to support an advanced protocol or network architecture.

C. THE SOFTWARE QUESTION

Minicomputer vendors typically provide a proprietary operating system, a selection of compilers, utilities and a limited amount of applications software. The operating system should, of course, support the intended application. If the system will be supporting transaction processing, it is imperative that the operating system support those tasks.

Similarly, if programs are to be developed in-house, it is important that the system support languages already being used by the programmers. The standard supported by languages under consideration should be examined carefully.

D. THE SERVICE FACTOR

Several areas of vendor support must be addressed during the buying cycle. The most obvious item is maintenance. The primary concern in this area is the availability and cost of a maintenance contract as well as the location of the nearest maintenance center, the scheduling of preventive maintenance, and the response time. One should also ascertain the vendor's policy concerning systems configured with third-party software and peripherals.

The buyers should determine a vendor's future marketing plan. If a vendor is involved in developing microcomputers to the exclusion of minicomputers, it is possible that the vendor will eventually vacate the minicomputer market. Such a move could leave the buyers without system support and would severly limit the availability of third-party peripherals and software.

The vendor's willingness to provide site planning, installation and initial system generation of services should be studied. These items are critical to the successful implementation of a new system [Ref. 11]

In this chapter we will not consider system needs separately, instead we will present an minicomputer selection model which considers hardware and software features by their weights. Thus this general evaluation method lets the buyer to set his own requirements by determining the weights of each factor. An evaluatin exanple will be presented for our system, but we can also use it whenever we would like to expand the system. Also, some type of minicomputers will be presented to give an idea about the system cost, under today's marketing conditions.

E. A SCALE FOR MINI COMPUTER SELECTION

Evaluating the minicomputers for selection needs a well-formed evaluation model. This model differs from other models in a sense that it addresses the relationships between objective measures (price, performance) and subjective (processor,memory size). Particular user needs and circumstances could require some variation from these suggested guidelines, however, the basic principles and methods which are presented offer enough flexibility to accommodate such changes. [Ref. 12]

F. A RIGOROUS EVALUATION MODEL

What is needed is an evaluation model which has four basic capabilities:

1. The ability to transform qualitative considerations into numerical units.
2. The ability to express both quantitative and qualitative ratings in the same units.
3. The ability to allow the decision-maker to explicitly his or her judgement as to the relative importance of qualitative and quantitative factors.

4. A capacity for performing sensitivity analysis on the results of the evaluation by the model.

The rigorous evaluation model (REM) has all of the above capabilities. The model expresses quantitative considerations as monetary costs, and transforms the total cost for each feasible alternative into a score between zero and one (the scores sum to one). For each qualitative factor, the model translates nominal ratings into numerical scores, and allows the decision-maker to express his view of the relative importance of the subjective criteria by weighing them (both the scores for each alternative on each criterion, and the associated weighted scores, sum to one). The decision-maker then selects a weighting scheme which represents his subjective view of the relative importance of the aggregate objective score, in relation to the aggregate subjective score.

The application of the model to computer systems is illustrated by the following example. In this example critical factors narrowed the number of minicomputers to be considered to five. Any computer its price range (for a minimum configuration) fell outside the range $80,000 to $150,000 was eliminated. The five qualifying computers were:

1. Data General Corporation, Eclipse MV/8000 II
    1. Multiuser-multiprogramming
    2. 1M bytes, 32 bits storage word, 32 bits transfer word.
    3. COBOL, BASIC, PL/1, PASCAL, APL, RPG, Language-C, FORTRAN 77, DG/L, SWAT available.
    4. OS: UNIX, AOS/VS, AOS/RT32

5. RS-232C, RS-449/422, 20/60 mA, RS-423 interfaces.

6. Asynchronous, synchronous, BSC, HDLC, X.25, SDLC, SNA, HASP II protocols.

7. Distribution: Vendor, Vendor maintenance.

8. 1983, Typical system: 1MB RAM, OS and application software. Price: $83,000-$240,000.

2. Dec Vax-11/730

1. Multiuser-multiprogramming

2. 1M to 5M bytes, 32 bits storage word, 32 bits transfer word. MOS memory, 24 users, virtual memory.

3. COBOL, BASIC, PL/1, BLISS 32, CORAL 66 Language-C, FORTRAN, DIBOL, MACRO assembly available.

4. OS: VAX/VMS

5. RS-232C, RS-449/422, 20/60 mA, RS-423 interfaces.

6. Asynchronous, synchronous, X.25, SNA, DDCMP protocols. Communications channels: 9

7. Distribution: Vendor, Vendor maintenance.

8. 1982, Typical system: 1MB RAM, 10MB disk, 121 MB fixed disk terminal, VAX/VMS, multifunction comm. ccntroller, Price: $28,500-$59,400

3. Ibm System/38 5381 Model 3

1. Multiuser-multiprogramming
2. 512k to 1.5 M bytes, 32 bits storage word, 32 bits transfer word.
3. MOSFET memory, 80 users, virtual memory.
4. BASIC, COBOL, RPG III, available.
5. OS: CPF
6. RS-232, CCITT V.35 interfaces. BSC, SDIC, SNA, protocols. Communication channels: 8
7. Distribution: Vendor, on-site maintenance,
8. 1980 price: $58,370 - $110,220. Lease (mo.) $2,403-$5,092

4. Prime Computer, Inc. Model 250-II

1. Multiuser-multiprogramming
2. 512K to 4M bytes, 32 bits storage word, 32 bits transfer word.
3. MOS memory, 32 users, virtual memory, cache 2KB
4. COBOL, BASIC, Pascal, RPG FORTRAN, assembly PL1/G available.
5. OS: PRIMOS
6. Asynchronous, X.25, BSC, HDLC, X.25, HASP, DPTX(3270),RJE emulation protocols. Communications channels:32, 1 DMA channel.
7. PRIMENET,RINGNET compatible.

8. Distribution: Vendor, on-site maintenance.

9. 1980, Typical system: 1MB RAM, 161 MB disk, 75 ips tape. CRT, 16 async lines, PRIMOS, Price: $78,000-$118,000

5. Wang Laboratories, Inc. VS 85

1. Multiuser-multiprogramming 1M to 4M bytes, 32 bits storage word, 32 bits transfer word.

2. MOS memory, 32-48 users, virtual memory, cache 32KB

3. COBOL, BASIC, PL/1, RPG, FORTRAN, assembly available.

4. OS: VS

5. BSC, IBM 3270, 2780/3780, 3274/3277,SNA, TTY, Siemens/BSVI ICL 7182 protocols.

6. WANGNET compatible.

7. Distribution: Vendor, vendor maintenance.

8. 1983, Price: $63,000.

Thirteen subjective factors were chosen. Each is discussed below, with a description of how each computer was rated.

- WORDSIZE AND CYCLE TIME Both affect computer speed and performance. The five computers were ranked 1 thru 5.
- PROCESSOR General capabilities, address and instruction lengths, number of registers. The computers were compared two at a time, the better receiving point of 1.

- **MEMORY EXPANSION** The maximum amount of random access memory (RAM) available through expansion. The number of KB were assigned as scores.

- **OTHER MEMORY FEATURES** Various features not included above. Each machine was assigned a score on a scale of 1 to 10.

- **MAXIMUM NUMBER OF CHANNELS** This factor will limit the I/O expansion. Each computer was ranked 1 thru 5.

- **NUMBER OF USERS** It is important in terms of system availability of potential user expansion.

- **DIRECT ACCESS MEMORY** The amount of RAM coming with the basic configuration. The number of KB RAM were assigned as scores.

- **OPERATING SYSTEM** The machine's operating system affects ease of use, and efficient use of the hardware. Operating systems were compared two at a time with a point given to the superior operating system.

- **ASSEMBLER** A poor assembler can limit some applications. The capabilities of the assembler are directly related to the hardware. A zero was assigned for an average or mediocre assembler and a one was assigned to a good assembler.

- **COMPILERS** The number and type of compilersavailable for that machine. Each system was assigned a rank from 1 to 5.

- **NETWORK SOFTWARE** Based on availability and quality a zero was assigned to poor software and a one was assigned to good software.

- **HARDWARE WITH MINIMUM CONFIGURATION** This factor rates the hardware that is included in the minimum configuration as packaged by the manufacturer. The hardware was compared two at a time, with a point to the better of the two.

- **OTHER FEATURES** Includes comments, which were not included in the other categories. A rank of 1 to 5 was assigned to each machine.

Each possible pair of the above factors was compared. A point was assigned to the more important factor of each comparison. The subjective weights were then computed by dividing the points for each factor by the total number of points. The resulted weights are presented in Table 1

TABLE 1

Subjective Factor Weights

NO	SUBJECTIVE FACTOR	WEIGHT
1	Wordsize and cycle time	.080
2	Processor	.128
3	Memory expansion	.110
4	Other memory features	.007
5	Maximum number of channels	.050
6	Number of users	.068
7	Direct access memory	.058
8	Operating system	.138
9	Assembler	.110
10	Compilers	.100
11	Network software	.077
12	Hardware with minimum configuration	.025
13	Other features	.011

The result of assigned weights in this manner was validated by ranking the subjective factors.

The scores for each computer on each factor were converted percentages. For each alternative Table 2 gives the score and percentage of subjective factor measure (SFM).

TABLE 2

Computation Of Subjective Factor Measures

DATAGEN	DEC	IBM	PRIME	WANG
2/.13	4/.27	5/.33	3/.20	1/.07
6/.33	1/.11	4/.44	0/.00	2/.22
2/.22	5000/.32	1500/.10	4000/.26	4000/.26
1000/.06	5/.15	7/.21	8/.24	9/.27
4/.12	24/.08	80/.26	32/.10	48/.15
128/.41	1000/.25	512/.13	512/.13	1000/.25
1000/.25	3/.30	2/.20	0/.00	1/.10
4/.40	1/.33	0/.00	1/.33	1/.33
0.00	2/.13	5/.33	2/.20	4/.27
1/.07	1/.20	1/.20	1/.20	1/.20
1/.20	4/.40	3/.30	1/.10	2/.20
0/.00	1/.07	2/.13	4/.27	5/.33
3/.20				
.17	.22	.22	.16	.23

The objective factor costs (OFC) were computed by taking the price of minimum configuration and adding five years of maintenance charges.

The objective factor measure (OFM) is computed based on the OFC. The subjective factor measure (SFM) is computed by summing the multiplication of each factor weight (SFW) by the respective weight for that factor given to each alternative.

The program then runs a sensitivity analysis by varying X from zero to 1 by .05 increments, where X is defined as the objective factor decision weight. The following formula provides the measure assigned to each alternative:

MEASURE= X (OFM(I)) + (1-X) (SFM(I))

The results of this program can be found in Table 3, whereas the following shows the sensitivity analysis of the data.

In Table 3, WANG received the highest subjective factor measure, followed by IBM and DEC. DEC received the highest objective measure, followed by IBM.

TABLE 3

Summary Of Objective and Subjective measures

TITLE	COST	MAINT	OPC	OPR	SPR
DATAGEN	83,000	60(572)	117320	.16	.17
DEC	50,000	60(344)	70640	.27	.22
IBM	58,000	60(400)	82000	.23	.22
PRIME	103,000	60(689)	144340	.13	.16
WANG	63,000	60(434)	89040	.21	.23

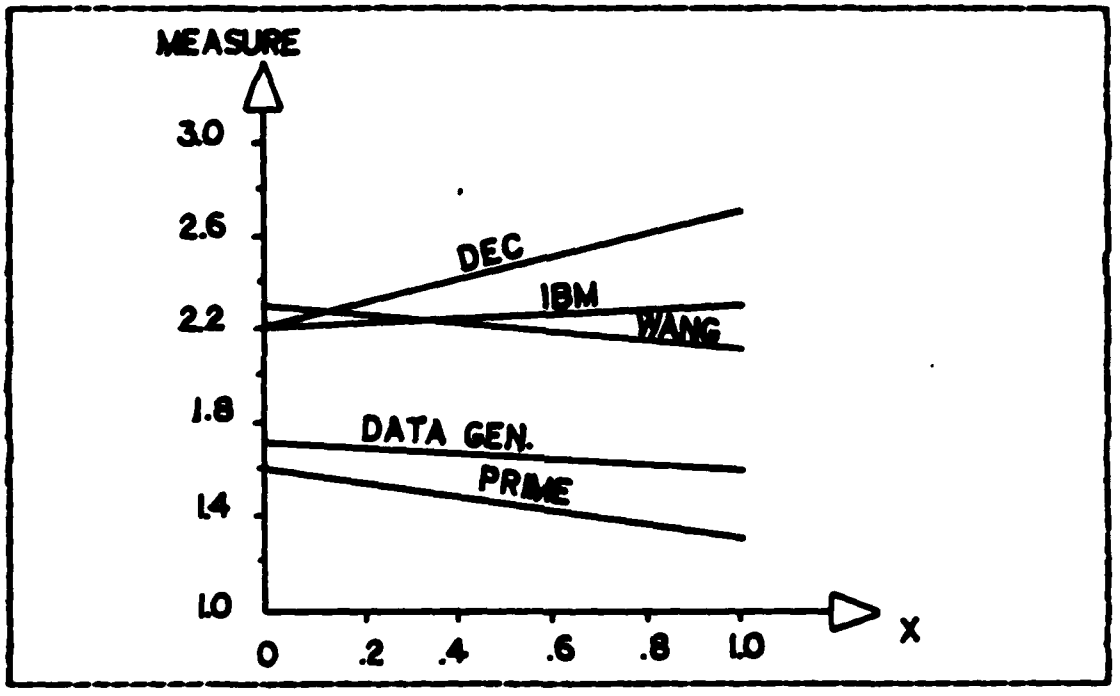


Figure 6.1 Subjectivity, Objectivity Relation.

Thus, if X were zero, one was basing his decision solely on subjectivity, the WANG computer would be selected. On the other extreme, if X were one, one was basing his decision on the objective measure, then the DEC would be selected.

Figure 6.1 shows the evaluation of five different computers. While x increasing, which means objectivity gains emphasis, on the other hand, while x decreasing, which means subjectivity gains emphasis. Thus, one can make his decision, by choosing an x value which is related with its economical constraints.

VII. CONCLUSIONS

1. In our system design considerations it is obvious that this type system can not be designed in ad hoc fashion. We need a central planning which at least looks five years ahead. Because the system cost will be very high and if we do not start with a detailed planning, we will have too many problems in the future and worse, to correct these errors may cost more than the original cost of the total system.
2. Database design is an important part of our overall system design. An inflexiable database will not let us to take advantage of using the computer power and network facilities. System expansion must be considered carefully. Dtabase back-up, recovery and integrity procedures are also very important issues. Auditing must be provided to integrate data and prevent the malicious accesses.
3. Network design or choosing a network architecture must be done carefully. Too many kinds of incompatible type of hardware and software products exist in marketplace. When we add the dollar factor it will be tougher job to choose these products. Also, networking needs a capability of expansion for further needs. Thus, it must be flexiable and permit the organization to change the system configuration, adding new users and so forth.
4. The actual cost of the system is very difficult to determine. The cost of minicomputers are fluctuating and many new products are coming in to the marketplace. Dollar consideration has tc be

balanced with the system needs, it shouldn't be considered as major factor. Since, the system's further benefits may balance costs. A structured system cost evaluation methodology must be kept during the selection process.

5. There are potentially large costs involved in training users and maintaining software. New system will require a group of programmer, analysts, to process software, trouble reports tests and changes, and maintain system documentation.

LIST OF REFERENCES

1. Martin, J. ,Design and Strategy For Distributed Data Processing, Prentice-Hall, Inc., 1981.

2. Hamner, Michael, and McLeod, Dennis."Database Description With SDM: A Semantic Database Model". In Transaction on Database Systems, Vol.6, No.3, Sep 1981.

3. Krcenke, D.,Database Processing, SRA Inc. , 1983.

4. Tanenbaum, A.S. ,Computer Networks, Prentice-Hall Inc., 1981.

5. Martin, J.,Systems Analysis For Data Transmission, Prentice-Hall Inc.,1972.

6. IBM General Information Manual on Binarysynchronous line control, Pcogh Keepsie, N.Y., n.d.

7. Martin, J.,Computer Networks and Distributed Processing, Prentice-Hall Inc.,1981.

8. McQuillan, J.,Adaptive Routing Algorithms For Distributed Computer Networks, Ph.D. Thesis, Div. of Engineering and Applied Sciences Hardward University, 1974.

9. Gerla, M., and Kleinrock, L.,"Topological Design of Distributed Computer Networks", IEEE Trans. Commun., Vol. COM-25, pp.48-60, 1977.

10. Hamming, R.W.,"Error Detecting And Correcting Codes", Bell System, Tech. J., Vol 29 pp.147-160, 1950.

11. Data Sources, Winter 83, Ziff-Davis publishing Co.

12. Ghandforoush, P.,"Model For Minicomputer Selection", The Journal of Systems Management, May 1981.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense technical information center Cameron Station Alexandria, Virginia 22314	2
2.	Library, code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Department Chairman, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93943	1
4.	Prof. Norman Lyons Code 54 LB Naval Postgraduate School Monterey, California 93943	1
5.	ICDS, Alan K. Johnson, Code 52 JN Computer Science Department Naval Postgraduate School Monterey, California 93943	1
6.	Turk Hava Kuvvetleri Komutanligi Per. Pst. D. Bsk. Bakanliklar/Ankara/TURKEY	3
7.	Hava Harp Okulu Komutanligi Kutuphane Yesilyurt/Istanbul/TURKEY	2
8.	Hava Harp Akademisi Komutanligi Kutuphane Ayazaga/Istanbul/TURKEY	2
9.	Engin Aytacer Kurtulus Mh. Inar Sk. no:75, Eskisehir/TURKEY	2
10.	Cem Gurdeniz Naval Postgraduate School SMC #2059 Monterey, California 93943	1
11.	Ugur Ozkan Naval Postgraduate School SMC #2451 Monterey, California 93943	1

END

FILMED

4-85

DTIC